

Teaching and Learning Guide for Web Design and Development



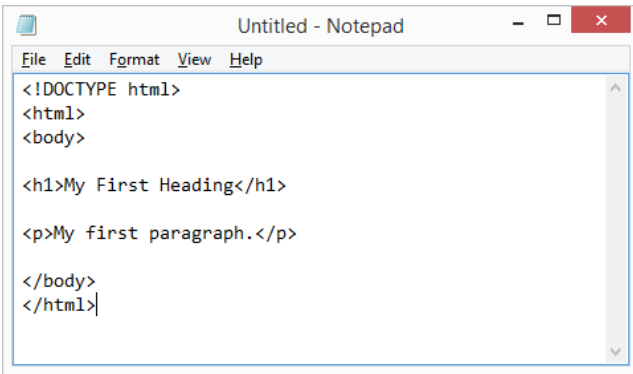
Contents

Teaching and Learning Guide for Web Design and Development.....	1
1 Elaborate World Wide Web	9
1.1 Definition of World Wide Web.....	9
1.2 Definition of website and webpage.....	9
1.2.1 Website	9
1.2.2 Webpage	9
1.3 Web 1.0, Web 2.0 and Web 3.0	9
1.4 Compare Static websites with Dynamic websites.....	10
1.4.1 Definition of Static website	10
1.4.2 Definition of Dynamic website.....	10
1.5 Architecture of static and dynamic Webpages.....	10
1.5.1 Architecture of a static Webpage.....	10
1.5.2 Architecture of a dynamic Webpage.....	10
1.6 Different types of websites.....	11
1.6.1 Web portal.....	11
1.6.2 Webmail.....	11
1.6.3 Social networking websites.....	11
1.6.4 Blogs	12
1.6.5 Forums	12
1.6.6 Wiki websites.....	12
1.6.7 Search engine websites	12
1.6.8 Community websites.....	12
1.6.9 News websites.....	13
1.7 Different web threats.....	13
1.7.1 Viruses	13
1.7.2 Computer worms.....	13
1.7.3 Trojans	13
1.7.4 Malware	13
1.7.5 SQL injection	13
1.7.6 Cross-site Scripting (XSS).....	13



1.7.7	Email Spam.....	14
1.7.8	Phishing.....	14
1.7.9	Denial-of-Service attack (DOS).....	14
1.7.10	Install antivirus software.....	14
1.7.11	Use antivirus to scan a computer for viruses and removal of viruses.....	14
1.8	Discuss shopping carts and ecommerce	14
1.8.1	Definition of E-commerce	14
1.8.2	What is Shopping cart.....	14
2	Design a Website	16
2.1	Definition of website designing	16
2.2	Types of Design	16
2.2.1	Fixed design	16
2.2.1	Liquid or Fluid design.....	16
2.2.2	Responsive design.....	16
2.3	Difference between web design and web development	17
2.3.1	What exactly is a Web Designer?	17
2.3.2	What exactly is a Web Developer?	17
2.4	Analyze different software to design a website	17
2.4.1	List of Web Design Software.....	17
2.4.2	Features and options of different designing software	17
2.4.3	Experience installation of different software to design a Website	17
2.4.4	Practice design in designing software like Adobe Photoshop, Macromedia Dreamweaver, Microsoft FrontPage etc.	17
2.5	Apply appropriate website Templates in your website	17
2.5.1	Explanation of a website template	17
2.5.2	Manage to searching free and paid website templates on internet.....	18
2.5.3	Downloading website templates to a computer	18
2.5.4	Learn how to test website templates using any web browser	18
2.5.5	Execute editing of different website templates in any web editor such as Macromedia Dreamweaver.....	18
3	Plan Website and Explain Software Development Life Cycle for Web Applications	19



3.1	Gather project requirements	19
3.2	Learn to execute storyboarding	19
3.3	Learn to develop timelines	20
3.4	Functional and non-functional requirements of a website	22
3.4.1	Functional Requirements	22
3.4.2	Non-Functional Requirements	22
3.5	Software Development Life Cycle for Web Applications	23
4	Develop Website using Client Side Scripting Languages	25
4.1	Write HTML code for a Website	25
4.1.1	Definition of Hypertext Mark-up Language (HTML)	25
4.1.2	HTML concepts	25
	Step 1: Open Notepad (PC)	25
	Step 2: Write Some HTML	25
		26
	Step 3: Save the HTML Page	26
	HTML Tag vs. Element	27
	Nested HTML Elements	27
	Core Attributes	29
	The Id Attribute	29
	The title Attribute	30
	The class Attribute	30
	The style Attribute	30
	Generic Attributes	31
	Headings Are Important	32
	Don't Forget the End Tag	32



Bold Text.....	33
Italic Text	33
Underlined Text.....	34
Strike Text.....	35
Monospaced Font	35
Superscript Text	36
Subscript Text	37
Inserted Text.....	37
Deleted Text	38
Larger Text.....	39
Smaller Text	39
Grouping Content.....	40
HTML Links - Syntax	42
HTML Links - The target Attribute	44
HTML Links - Image as Link	44
Chapter Summary.....	46
The HTML <head> Element.....	46
The HTML <style> Element.....	47
The HTML <link> Element	47
Setting the Viewport.....	48
HTML head Elements	50
Images on another Server.....	53
Animated Images.....	53
HTML Table - Adding a Border	55
HTML Table - Collapsed Borders	55
th { text-align: left; }	56
HTML Table - Adding Border Spacing	56
table { border-spacing: 5px; }	56
HTML Table - Cells that Span Many Columns	56
HTML Table - Cells that Span Many Rows	57
HTML Table - Adding a Caption.....	57



A Special Style for One Table.....	58
Chapter Summary.....	59
Unordered HTML List	59
Unordered HTML List - Choose List Item Marker.....	60
Value.....	60
Description.....	60
disc.....	60
Sets the list item marker to a bullet (default).....	60
circle	60
Sets the list item marker to a circle.....	60
square.....	60
Sets the list item marker to a square	60
none	60
The list items will not be marked	60
Example - Disc	60
Ordered HTML List	61
Ordered HTML List - The Type Attribute	61
Type	61
Description.....	61
type="1".....	61
The list items will be numbered with numbers (default).....	61
Type="A"	61
The list items will be numbered with uppercase letters.....	61
Type="a"	61
The list items will be numbered with lowercase letters.....	61
Type="I"	61
The list items will be numbered with uppercase roman numbers	61
Type="i"	61
The list items will be numbered with lowercase roman numbers	61
Numbers:	61
Lowercase Letters:.....	62



HTML Description Lists	62
Nested HTML Lists.....	63
Horizontal Lists	63
Chapter Summary.....	64
Block-level Elements	65
Inline Elements.....	65
HTML Layout Elements	65
HTML Layout Techniques.....	66
Form Attributes	67
HTML Form Controls	67
Text Input Controls.....	68
Button Controls.....	75
Hidden Form Controls	76
Iframe Syntax	77
Iframe - Set Height and Width	77
<iframe src="demo_iframe.htm" height="200" width="300"></iframe>	77
Iframe - Remove the Border.....	77
Iframe - Target for a Link	77
HTML iframe Tag.....	78
Tag.....	78
Description	78
<iframe>	78
Defines an inline frame	78
Input Type Text.....	78
Input Type Password.....	79
Input Type Submit.....	79
Input Type Reset	80
Input Type Radio.....	81
Input Type Checkbox	81
Input Type Button	81
HTML5 Input Types	82



Input Type Number	82
Input Type Date.....	83
Input Type Color	83
Input Type Range	83
Input Type Month.....	84
Input Type Week.....	84
Input Type Time.....	84
Input Type Datetime-local	84
Input Type Email.....	85
Input Type Search	85
Input Type Tel.....	85
Input Type Url.....	85
Input Restrictions	86
Multimedia Formats	87
Sound Formats.....	88
Audio on the Web	89
Playing Videos in HTML.....	89



1 Elaborate World Wide Web

1.1 Definition of World Wide Web

An information system on the Internet which allows documents to be connected to other documents by hypertext links, enabling the user to search for information by moving from one document to another.

1.2 Definition of website and webpage

1.2.1 Website

A location connected to the Internet that maintains one or more web pages.

1.2.2 Webpage

A hypertext document connected to the World Wide Web.

1.3 Web 1.0, Web 2.0 and Web 3.0

Web 1.0

It is the “readable” phrase of the World Wide Web with flat data. In Web 1.0, there is only limited interaction between sites and web users. Web 1.0 is simply an information portal where users passively receive information without being given the opportunity to post reviews, comments, and feedback.

Web 2.0

It is the “writable” phrase of the World Wide Web with interactive data. Unlike Web 1.0, Web 2.0 facilitates interaction between web users and sites, so it allows users to interact more freely with each other. Web 2.0 encourages participation, collaboration, and information sharing. Examples of Web 2.0 applications are YouTube, Wiki, Flickr, Facebook, and so on.

Web 3.0

It is the “executable” phrase of Word Wide Web with dynamic applications, interactive services, and “machine-to-machine” interaction. Web 3.0 is a semantic web which refers to the future. In Web 3.0, computers can interpret information like humans and intelligently generate and distribute useful content tailored to the needs of users. One example of Web 3.0 is Tivo, a digital video recorder. Its recording program can search the web and read what it finds to you based on your preferences.



1.4 Compare Static websites with Dynamic websites

1.4.1 Definition of Static website

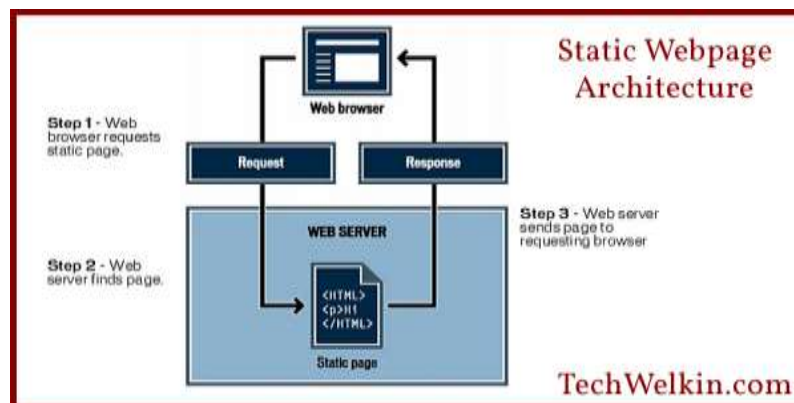
A static website contains Web pages with fixed content. Each page is coded in HTML and displays the same information to every visitor. Static sites are the most basic type of website and are the easiest to create. Unlike dynamic websites, they do not require any Web programming or database design. A static site can be built by simply creating a few HTML pages and publishing them to a Web server.

1.4.2 Definition of Dynamic website

Dynamic websites contain Web pages that are generated in real-time. These pages include Web scripting code, such as PHP or ASP. When a dynamic page is accessed, the code within the page is parsed on the Web server and the resulting HTML is sent to the client's Web browser.

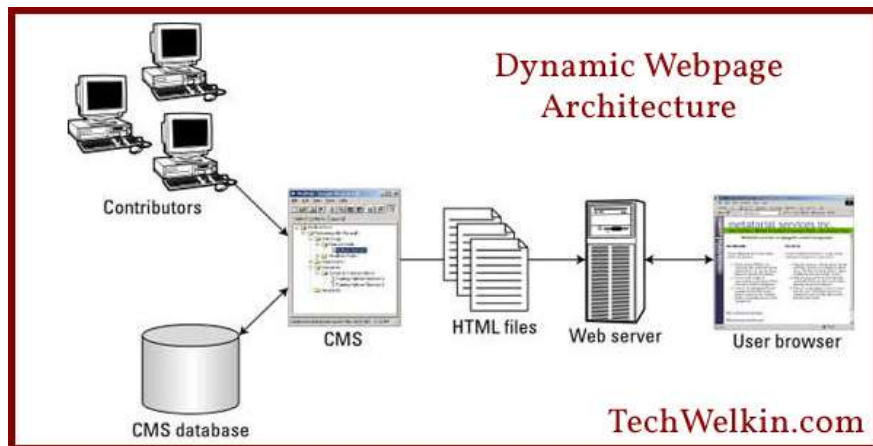
1.5 Architecture of static and dynamic Webpages

1.5.1 Architecture of a static Webpage



1.5.2 Architecture of a dynamic Webpage





1.6 Different types of websites

1.6.1 Web portal

A Web portal is most often a specially designed web site that brings information together from diverse sources in a uniform way. Usually, each information source gets its dedicated area on the page for displaying information (a portlet); often, the user can configure which ones to display.

Examples

- <http://australia.gov.au> for Australia.
- <http://www.pakistan.gov.pk> for Pakistan.
- <http://www.gov.lk> for Sri Lanka.
- <http://gov.uk> for citizens & businesslink.gov.uk for businesses in the United Kingdom.
- <http://india.gov.in> for India.

1.6.2 Webmail

Webmail are web-based email accounts. These are usually free email accounts that are operated from a website. Examples include Hotmail, Gmail and Yahoo Mail. Webmail allows the users to access their emails as long as they have access to an Internet connection and a web browser.

1.6.3 Social networking websites

Social networking is the use of internet-based social media programs to make connections with friends, family, classmates, customers and clients. Social networking can occur for social purposes, business purposes or both through sites such as Facebook, Twitter, LinkedIn, Classmates.com and Yelp.



1.6.4 Blogs

A website, similar to an online journal, that includes chronological entries made by individuals. The word blog was derived from the combination of the word web and log. Blogs typically focus on a specific subject (Economy, entertainment news, etc.) and provide users with forums (or a comment area) to talk about each posting. Many people use blogs as they would a personal journal or diary.

1.6.5 Forums

An Internet forum, or message board, is an online discussion site where people can hold conversations in the form of posted messages. They differ from chat rooms in that messages are often longer than one line of text, and are at least temporarily archived. Also, depending on the access level of a user or the forum set-up, a posted message might need to be approved by a moderator before it becomes visible.

Forums have a specific set of jargon associated with them; e.g., a single conversation is called a "thread", or topic.

1.6.6 Wiki websites

A wiki is a website that provides collaborative modification of its content and structure directly from the web browser. In a typical wiki, text is written using a simplified markup language (known as "wiki markup"), and often edited with the help of a rich-text editor. A wiki is run using wiki software, otherwise known as a wiki engine.

1.6.7 Search engine websites

Search engines are programs that search documents for specified keywords and return a list of the documents where the keywords were found. A search engine is really a general class of programs; however, the term is often used to specifically describe systems like Google, Bing and Yahoo! Search that enable users to search for documents on the World Wide Web.

1.6.8 Community websites

An online community is a virtual community whose members interact with each other primarily via the Internet. For many, online communities may feel like home, consisting of a "family of invisible friends. "Those who wish to be a part of an online community usually have to become a member via a specific site and necessarily need an internet connection. An online community can act as an information system where members can post, comment on discussions, give advice or collaborate. Commonly, people communicate through social networking sites, chat rooms, forums, e-mail lists and discussion boards. People may also join online communities through video games, blogs and virtual worlds.



1.6.9 News websites

An online newspaper is the online version of a newspaper, either as a stand-alone publication or as the online version of a printed periodical. Going online created more opportunities for newspapers, such as competing with broadcast journalism in presenting breaking news in a timelier manner.

1.7 Different web threats

1.7.1 Viruses

A computer virus is a type of malicious software program ("malware") that, when executed, replicates by reproducing itself (copying its own source code) or infecting other computer programs by modifying them. Infecting computer programs can include as well, data files, or the "boot" sector of the hard drive. When this replication succeeds, the affected areas are then said to be "infected" with a computer virus.

1.7.2 Computer worms

A computer worm is a self-replicating computer program that penetrates an operating system with the intent of spreading malicious code. Worms utilize networks to send copies of the original code to other computers, causing harm by consuming bandwidth or possibly deleting files or sending documents via email.

1.7.3 Trojans

In computing, Trojan horse, or Trojan, is any malicious computer program which is used to hack into a computer by misleading users of its true intent.

1.7.4 Malware

Malware, short for malicious software, is any software used to disrupt computer or mobile operations, gather sensitive information, gain access to private computer systems, or display unwanted advertising.

1.7.5 SQL injection

SQL Injection (SQLi) refers to an injection attack wherein an attacker can execute malicious SQL statements (also commonly referred to as a malicious payload) that control a web application's database server.

1.7.6 Cross-site Scripting (XSS)



Cross-site scripting (XSS) is a type of computer security vulnerability typically found in web applications. XSS enables attackers to inject client-side scripts into web pages viewed by other users.

1.7.7 Email Spam

Email spam—also known as junk email—is a type of electronic spam where unsolicited messages are sent by email. Many email spam messages are commercial in nature but may also contain disguised links that appear to be for familiar websites but in fact lead to phishing web sites or sites that are hosting malware.

1.7.8 Phishing

Phishing is the attempt to obtain sensitive information such as usernames, passwords, and credit card details (and, indirectly, money), often for malicious reasons, by disguising as a trustworthy entity in an electronic communication.

1.7.9 Denial-of-Service attack (DOS)

Short for denial-of-service attack, a type of attack on a network that is designed to bring the network to its knees by flooding it with useless traffic.

1.7.10 Install antivirus software.

1.7.11 Use antivirus to scan a computer for viruses and removal of viruses.

The above tasks will be performed in lab.

1.8 Discuss shopping carts and ecommerce

1.8.1 Definition of E-commerce

E-commerce (electronic commerce or EC) is the buying and selling of goods and services, or the transmitting of funds or data, over an electronic network, primarily the internet. These business transactions occur either as business-to-business, business-to-consumer, consumer-to-consumer or consumer-to-business. The terms e-commerce and e-business are often used interchangeably.

1.8.2 What is Shopping cart

A shopping cart is a piece of software that acts as an online store's catalog and ordering process. Typically, a shopping cart is the interface between a company's Web site and its deeper infrastructure, allowing consumers to select merchandise; review what they have selected; make necessary modifications or additions; and purchase the merchandise.



Shopping carts can be sold as independent pieces of software so companies can integrate them into their own unique online solution, or they can be offered as a feature from a service that will create and host a company's e-commerce site.



2 Design a Website

2.1 Definition of website designing

Web design is the process of creating websites. It encompasses several different aspects, including webpage layout, content production, and graphic design. While the terms web design and web development is often used interchangeably, web design is technically a subset of the broader category of web development.

Websites are created using a markup language called HTML. Web designers build webpages using HTML tags that define the content and metadata of each page. The layout and appearance of the elements within a webpage are typically defined using CSS, or cascading style sheets. Therefore, most websites include a combination of HTML and CSS that defines how each page will appear in a browser.

2.2 Types of Design

2.2.1 Fixed design

Fixed web pages have a set width that won't alter when the browser is resized, no matter what device the website is being viewed on. On smaller devices, such as smartphones or tablets, content is harder to view and can be annoying for users due to the need to scroll horizontally to view the rest of the content on a page, or continuously "pinch and expand" to zoom into the page's text. When the browser on a screen is enlarged or reduced, images and text may visually fall apart on the screen.

2.2.1 Liquid or Fluid design

When resizing the browser, the content on the page spreads itself out to fill the width of the browser when expanded, hence the term liquid design, and will look enlarged or as though is has shrunk. The columns containing the content on the webpage are built using percentages, rather than fixed columns used in fixed design, therefore the columns increase or decrease in size relative to each other.

2.2.2 Responsive design

This approach aims to make website viewing easier by displaying websites on different devices in forms that are easy to read and navigate. This avoids the user from having to resize, pan or scroll through the webpage to read the website's content. Websites created using responsive design are designed to display different content as the browser is expanded or reduced to predetermined sizes. For example, when the browser size is reduced to 70% of its maximum width, the webpage may have been set to display only two columns on the



screen rather than three. When the browser is expanded past 70% of the screen, the third column of content will return to the screen.

2.3 Difference between web design and web development

2.3.1 What exactly is a Web Designer?

The best Web Designers are of the creative type. They have a knack for getting inside of their clients' heads and realizing their clients' vision. They take this vision and masterfully convert it into an aesthetically pleasing, artistic design that aims to impress millions of potential viewers. Some designers study typography, user interface design, and usability.

2.3.2 What exactly is a Web Developer?

Web Developers are usually more technical in nature. They tend to have excellent problem solving skills and are generally good at math. On a daily basis, a developer will write code in five or more different languages including (X)HTML, CSS, JavaScript, [PHP, ASP, ColdFusion, Ruby, Python, Perl, etc.], and some flavor of SQL.

2.4 Analyze different software to design a website

2.4.1 List of Web Design Software

1. Wix.com
2. Weebly
3. Squarespace

2.4.2 Features and options of different designing software

2.4.3 Experience installation of different software to design a Website

2.4.4 Practice design in designing software like Adobe Photoshop, Macromedia Dreamweaver, Microsoft FrontPage etc.

Above tasks will be performed in lab.

2.5 Apply appropriate website Templates in your website

2.5.1 Explanation of a website template

2.5.1.1 What are Website Templates?

A website template (or web template) is a pre-designed webpage or set of HTML webpages that anyone can use to "plug-in" their own text content and images into to create a website. Usually built with HTML and CSS code, website templates allow anyone to setup a website.



2.5.2 Manage to searching free and paid website templates on internet

2.5.3 Downloading website templates to a computer

2.5.4 Learn how to test website templates using any web browser

2.5.5 Execute editing of different website templates in any web editor such as Macromedia Dreamweaver

Above tasks will be performed during lab.



3 Plan Website and Explain Software Development Life Cycle for Web Applications

3.1 Gather project requirements

Website Requirements

Website requirements are a list of necessary functions, capabilities, or characteristics related to your website and the plans for creating it.

Types of Requirements

There are many different types of requirements documentation. At a higher level, most can fall within one of the following categories:

Business Requirements define the objectives and what problems the stakeholder intends to solve with the product.

User Requirements describe how user expectations and how they will interact with the product. Use the features, functions, and content described in your scenarios to develop your requirements.

Functional Requirements provide details of how a product should behave and specify what is needed for development.

Quality-of-Service Requirements detail what characteristics a product must maintain in order to maintain its effectiveness and any constraints.

Implementation Requirements are used to detail changes in process, team roles, migration from one system to another, etc.

Using Website Requirements

Website requirements only tell you what your website must have and what it must allow users to do.

3.2 Learn to execute storyboarding

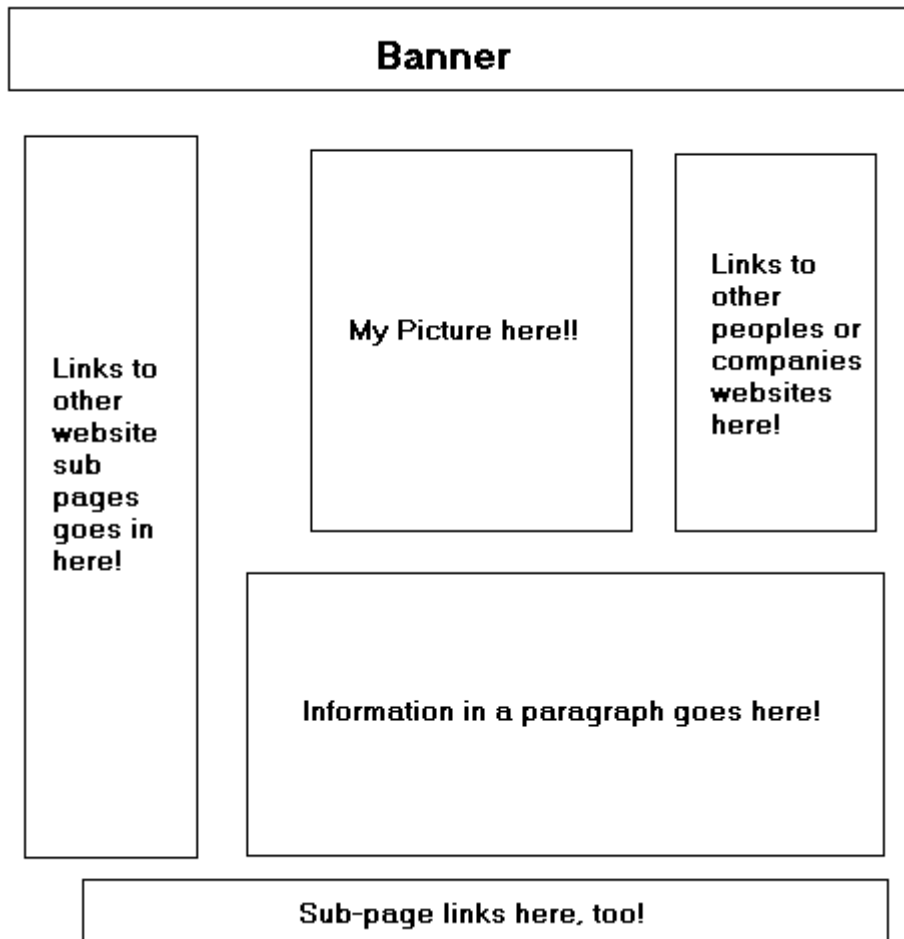
Website Storyboarding

When thinking about some building blocks, get a sheet of paper and start to draw your website. Start with the homepage, the homepage is one of the most important pages. You have 5 seconds to attract your visitor your site. If your site is too hard to use, nobody is going to stay.



Make it simple. When you have completed the first storyboard, try to think like a first time visitor. Would you stop and leave? Or would you click on a link? If you would click a link, then you have a great storyboard design.

You can quickly draw a quick sketch of a storyboard in seconds. Here is a storyboard example:



3.3 Learn to develop timelines

Website Timeline: A Plan for Success

Category: Design, Strategy

The two most common questions that we are asked when it comes to producing a new website revolve around price and time. “How much does a new website cost?” “How long does it take to build a new website?” As with most projects, the answers can vary dramatically as it depends on the size and scope of the project. We establish milestones for every web project and ensure that we hit the goals at each stage before moving to the next. The following list outlines the milestones involved when developing a new website.



Discovery and planning (2 to 10 weeks) – Areas covered at this stage may include research; discovery sessions; writing of creative brief; writing of a technical briefs or more detailed architecture, requirements and definitions document; site architecture plan; wireframes and sitemap creation.

Design (4 to 12 weeks) –It is at this stage that design mock ups are created and presented for feedback to establish a design that meets the goals of the new website. Design mockups may include homepage designs, multiple internal page designs, desktop views and mobile views.

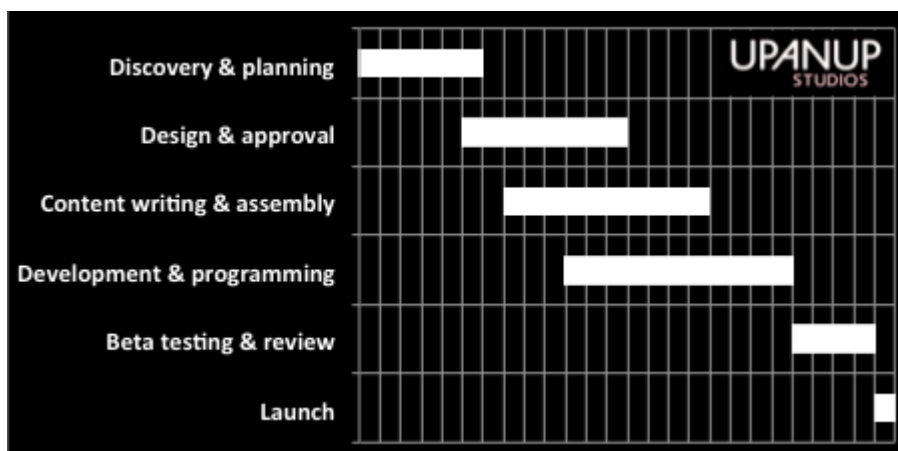
Content writing and assembly (5 to 15 weeks) – The value of good content to any website cannot be underestimated. As important as a good design, quality content can set a website apart. Content writing and assembly involves establishing key messages and calls-to-action, porting over content from existing sites, writing new text, editing text, creating headlines, page titles, captions and SEO optimized text, headlines, descriptions and tags.

Development and Programming (6 to 15 weeks) – This is where the website comes to life and all the elements of the site come together to produce a high quality website. Timeline for this phase can stretch even longer.

Beta testing and review (2 to 6 weeks) – Once the site has been developed it then goes to a beta testing stage. The opportunity is given to the client to share the site with the necessary stakeholders for review and feedback in a password-protected location. Testing is also conducted during this stage to ensure the site is optimized across multiple browsers, screens and mobile devices and ensure the site is performing as it should prior to launch.

Site launch (1 day to 2 weeks) – Upon approval from the beta stage the new website is then launched with no interruption to the current site. The launch process can be simple and quick, or can be more involved depending on what is involved.

Maintenance and Enhancements (ongoing) – Once a site has gone live, there is always room for ongoing improvements and updates.



3.4 Functional and non-functional requirements of a website

Before creating any website its common practice to visualize the layout, the design and all the features you intend to incorporate. In addition, you think about how users will interact with each page and how the site should perform (behavior, load time etc.). In software engineering, establishing a list of requirements for a program is referred to as developing the functional and nonfunctional requirements.

3.4.1 Functional Requirements

In a nut shell, functional requirements describe what the software / website should do (the functions). Think about the core operations.

Because the “functions” are established before development, functional requirements should be written in the future tense. In developing the web application for the auto car shop, some of the functional requirements could include:

- The web application shall accept customer orders
- The web application shall be able to cash a sale
- The web application shall produce a receipt detailing a customers’ purchase information and include name of customer, items purchased, cost of each item and total cost
- The web application shall be able to produce weekly, monthly and yearly reports about sales

3.4.2 Non-Functional Requirements

Non-functional requirements are not concerned with the functions of the system. Instead, they look at the criteria to which the software or website is expected to conform to. Non-functional requirements can include things like response time and reliability. It can also be closely tied to user satisfaction.

Some non-functional requirements for the auto shop application could include:



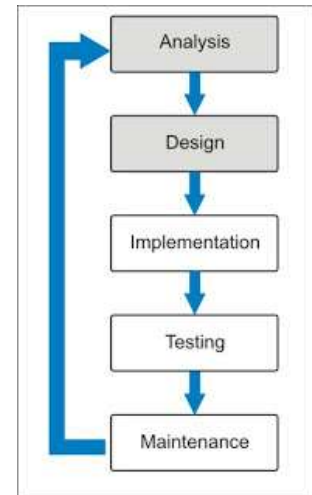
- The web application shall be easy to use by all employees including sales representatives and managers
- The web application shall be available in several languages
- The web application shall allow several sales to be made at the same time without downgrading performance



Notice how each requirement is not related to an operation or what the application should do. Instead, the main focus in this case is the ease of use and performance. That's the idea of non-functional requirements.

3.5 Software Development Life Cycle for Web Applications

A successful product development with customer satisfaction aspects may sound easier said than done as it involves quite a bit of time, energy and brains. Consulting specialist web development services would be a good idea to help you get your job done accurately in given time frame. Many models and techniques are followed for software development these days but most widely used is named as Software Development Life Cycle (SDLC) or Waterfall Model. It comprises mainly of following seven steps.



1. Project Planning:

This level involves determining the project goal and running a feasibility study amongst the client and web development services, taking into consideration various factors like project cost, equipment cost, practicality etc.

2. System Requirement Analysis

Refinement of project goals into defined functions and operations of the proposed application through intensive discussion between web development services and the client is achieved through this step.

3. System Design

Documentation of various details like operations and functions such as screen layouts, process diagrams and other documentation are done here.

4. Implementation

This is where the expertise of web development services are needed the most when actual back end coding is done.

5. Testing

In this phase the product is put through various testing environments and tools designed and used by web development services to make the product to remove its bugs and errors to ensure harmonious execution.

6. Acceptance and Deployment



Finally the web development services deploy and install the system after getting formally approved by the client.

7. Maintenance

The web development services not only make sure the installation of the application but they are also responsible for subsequent maintenance and upgrading if and when needed.



4 Develop Website using Client Side Scripting Languages

4.1 Write HTML code for a Website

4.1.1 Definition of Hypertext Mark-up Language (HTML)

HTML stands for Hypertext Markup Language, and it is the most widely used language to write Web Pages.

- Hypertext refers to the way in which Web pages (HTML documents) are linked together. Thus, the link available on a webpage is called Hypertext.
- A text document with tags that tell a Web browser how to structure it to display.

Originally, HTML was developed with the intent of defining the structure of documents like headings, paragraphs, lists, and so forth to facilitate the sharing of scientific information between researchers. Now, HTML is being widely used to format web pages with the help of different tags available in HTML language.

4.1.2 HTML concepts

4.1.2.1 HTML Editors

Web pages can be created and modified by using professional HTML editors, however, for learning HTML we recommend a simple text editor like Notepad (PC) or Text Edit (Mac). We believe using a simple text editor is a good way to learn HTML.

Follow the four steps below to create your first web page with Notepad or Text Edit.

Step 1: Open Notepad (PC)

Windows 8 or later:

Open the **Start Screen** (the window symbol at the bottom left on your screen). Type **Notepad**.

Step 2: Write Some HTML

Write or copy some HTML into Notepad

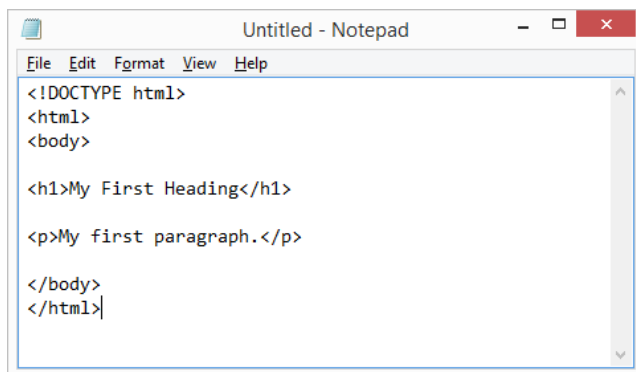


```
<!DOCTYPE html>
<html>
<body>

<h1>My First Heading</h1>

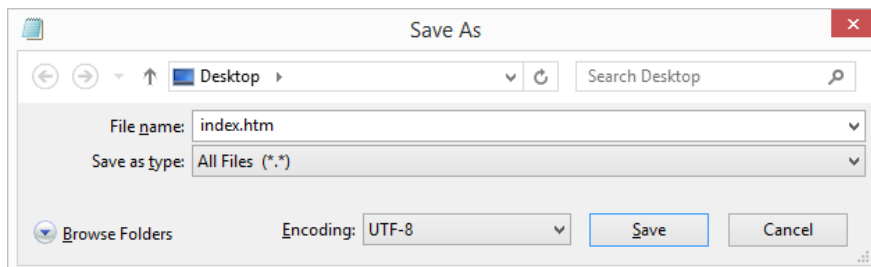
<p>My first paragraph.</p>

</body>
</html>
```



Step 3: Save the HTML Page

Save the file on your computer. Select **File > Save as** in the Notepad menu. Name the file "**index.html**".



Step 4: View the HTML Page in Your Browser

Open the saved HTML file in your favorite browser (double clicks on the file, or right-click - and choose "Open with").





4.1.2.2 HTML Elements

An HTML element is defined by a starting tag. If the element contains other content, it ends with a closing tag, where the element name is preceded by a forward slash as shown below with few tags:

Start Tag	Content	End Tag
<code><p></code>	This is paragraph content.	<code></p></code>
<code><h1></code>	This is heading content.	<code></h1></code>
<code><div></code>	This is division content.	<code></div></code>
<code>
</code>		

So here `<p>....</p>` is an HTML element, `<h1>...</h1>` is another HTML element. There are some HTML elements which don't need to be closed, such as `<img.../>`, `<hr/>` and `
` elements. These are known as void elements.

HTML Tag vs. Element

An HTML element is defined by a starting tag. If the element contains other content, it ends with a closing tag. For example, `<p>` is starting tag of a paragraph and `</p>` is closing tag of the same paragraph but `<p>. This is paragraph</p>` is a paragraph element.

Nested HTML Elements

It is very much allowed to keep one HTML element inside another HTML element:



```

<!DOCTYPEhtml>

<html>

<head>

<title>Nested Elements Example</title>

</head>

<body>

<h1>This is <i>italic</i> heading</h1>

<p>This is <u>underlined</u> paragraph</p>

</body>

</html>

```

This will display the following result:



4.1.2.3 HTML Attributes

An attribute is used to define the characteristics of an HTML element and is placed inside the element's opening tag. All attributes are made up of two parts: a name and a value:

- The name is the property you want to set. For example, the paragraph <p> element in the example carries an attribute whose name is align, which you can use to indicate the alignment of paragraph on the page.
- The value is what you want the value of the property to be set and always put within quotations. The below example shows three possible values of align attribute: left, center and right.

Attribute names and attribute values are case-insensitive. However, the World Wide Web Consortium (W3C) recommends lowercase attributes/attribute values in their HTML 4 recommendation.



```
<!DOCTYPE html>

<html>

<head>

  <title>Align Attribute Example</title>

</head>

<body>

  <p align="left">This is left aligned</p>

  <p align="center">This is center aligned</p>

  <p align="right">This is right aligned</p>

</body>

</html>
```

This will display the following result:



Core Attributes

The four core attributes that can be used on the majority of HTML elements (although not all) are:

- Id
- Title
- Class
- Style

The Id Attribute

The id attribute of an HTML tag can be used to uniquely identify any element within an HTML page. There are two primary reasons that you might want to use an id attribute on an element:

- If an element carries an id attribute as a unique identifier, it is possible to identify just that element and its content.
- If you have two elements of the same name within a Web page (or style sheet), you can use the id attribute to distinguish between elements that have the same name.



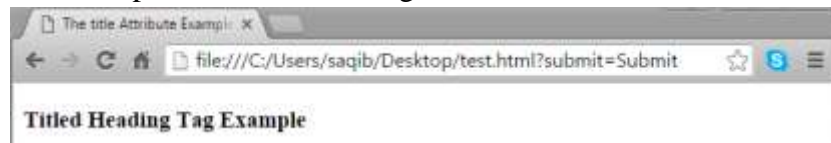
The title Attribute

The title attribute gives a suggested title for the element. The syntax for the title attribute is similar as explained for id attribute:

The behavior of this attribute will depend upon the element that carries it, although it is often displayed as a tooltip when cursor comes over the element or while the element is loading.

```
<!DOCTYPE html>
<html>
<head>
  <title>The title Attribute Example</title>
</head>
<body>
  <h3 title="Hello HTML!">Titled Heading Tag Example</h3>
</body>
</html>
```

This will produce the following result:



The class Attribute

The class attribute is used to associate an element with a style sheet, and specifies the class of element. The value of the attribute may also be a space-separated list of class names. For example:

```
class="className1 className2 className3"
```

The style Attribute

The style attribute allows you to specify Cascading Style Sheet (CSS) rules within the element.

```
<!DOCTYPE html>

<html>

<head>

<title>The style Attribute</title>

</head>

<body>

<p style="font-family: arial; color: #FF0000;">Some text...</p>

</body>

</html>
```



This will produce the following result:



Generic Attributes

Here's a table of some other attributes that are readily usable with many of the HTML tags.

Attribute	Options	Function
Align	right, left, center	Horizontally aligns tags
valign	top, middle, bottom	Vertically aligns tags within an HTML element.
b bgcolor	numeric, hexadecimal, RGB values	Places a background color behind an element
background	URL	Places a background image behind an element
id	User Defined	Names an element for use with Cascading Style Sheets.
class	User Defined	Classifies an element for use with Cascading Style Sheets.
height	Numeric Value	Specifies the height of tables, images, or table cells.
title	User Defined	"Pop-up" title of the elements.

We will see related examples as we will proceed to study other HTML tags.

4.1.2.4 HTML Headings

Headings are defined with the <h1> to <h6> tags.

<h1> defines the most important heading. <h6> defines the least important heading.

Example

```
<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<h3>This is heading 3</h3>
<h4>This is heading 4</h4>
<h5>This is heading 5</h5>
<h6>This is heading 6</h6>
```



Headings Are Important

Search engines use the headings to index the structure and content of your web pages. Users skim your pages by its headings. It is important to use headings to show the document structure. <h1> headings should be used for main headings, followed by <h2> headings, then the less important <h3>, and so on.

Note: Browsers automatically add some white space (a margin) before and after a heading.

Note: Use HTML headings for headings only. Don't use headings to make text **BIG** or **bold**.

4.1.2.5 HTML Paragraphs

The HTML <p> element defines a **paragraph**:

Note: Browsers automatically add some white space (a margin) before and after a paragraph.

```
<p>This is a paragraph.</p>  
<p>This is another paragraph.</p>
```

Don't Forget the End Tag

Most browsers will display HTML correctly even if you forget the end tag:

Example

```
<p>This is a paragraph.  
<p>This is another paragraph.
```

The example above will work in most browsers, but do not rely on it.

Note: Dropping the end tag can produce unexpected results or errors.



4.1.2.6 HTML Formatting

If you use a word processor, you must be familiar with the ability to make text bold, italicized, or underlined; these are just three of the ten options available to indicate how text can appear in HTML and XHTML.

Bold Text

Anything that appears within `...` element, is displayed in bold as shown below:

Example:

```
<!DOCTYPE html>

<html>

<head>

    <title>Bold Text Example</title>

</head>

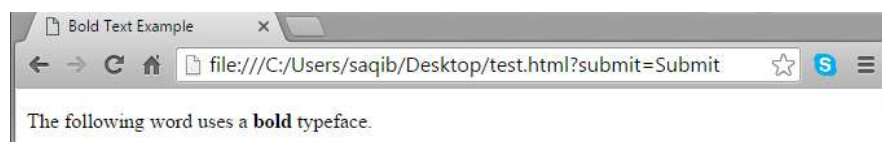
<body>

    <p>The following word uses a <b>bold</b> typeface.</p>

</body>

</html>
```

This will produce the following result:



Italic Text

Anything that appears within `<i>...</i>` element is displayed in italicized as shown below:

Example:

```
<!DOCTYPE html>

<html>

<head>

    <title>Italic Text Example</title>

</head>

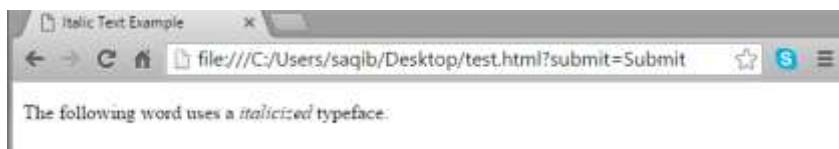
<body>

    <p>The following word uses a <i>italicized</i> typeface.</p>

</body>

</html>
```

This will produce the following result:



Underlined Text

Anything that appears within <u>...</u> element, is displayed with underline as shown below:

Example:

```
<!DOCTYPE html>

<html>

<head>

    <title>Underlined Text Example</title>

</head>

<body>

    <p>The following word uses a <u>underlined</u> typeface.</p>

</body>

</html>
```

This will produce the following result:





Strike Text

Anything that appears within `<strike>...</strike>` element is displayed with strikethrough, which is a thin line through the text as shown below:

Example:

```
<!DOCTYPE html>

<html>

<head>

    <title>Strike Text Example</title>

</head>

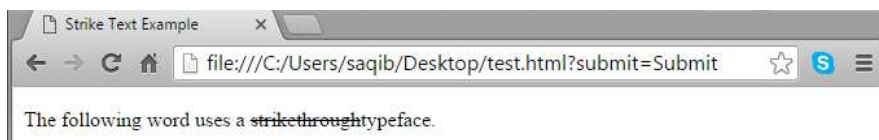
<body>

    <p>The following word uses a <strike>strikethrough</strike>typeface.</p>

</body>

</html>
```

This will produce the following result:



Monospaced Font

The content of a `<tt>...</tt>` element is written in monospaced font. Most of the fonts are known as variable-width fonts because different letters are of different widths (for example, the letter 'm' is wider than the letter 'i'). In a monospaced font, however, each letter has the same width.



```
<!DOCTYPE html>

<html>

    <head>

        <title>Monospaced Font Example</title>

    </head>

    <body>

        <p>The following word uses a <tt>monospaced</tt> typeface.</p>

    </body>

</html>
```

This will produce the following result:



Superscript Text

The content of a `^{...}` element is written in superscript; the font size used is the same size as the characters surrounding it but is displayed half a character's height above the other characters.

Example:

```
<!DOCTYPE html>

<html>

    <head>

        <title>Superscript Text Example</title>

    </head>

    <body>

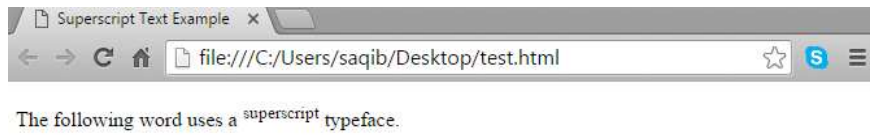
        <p>The following word uses a <sup>superscript</sup> typeface.</p>

    </body>

</html>
```

This will produce the following result:





Subscript Text

The content of a `_{...}` element is written in subscript; the font size used is the same as the characters surrounding it, but is displayed half a character's height beneath the other characters.

Example:

```
<!DOCTYPE html>

<html>

    <head>

        <title>Subscript Text Example</title>

    </head>

    <body>

        <p>The following word uses a <sub>subscript</sub> typeface.</p>

    </body>

</html>
```

This will produce the following result:



Inserted Text

Anything that appears within `<ins>...</ins>` element is displayed as inserted text.

Example:



```
<!DOCTYPE html>

<html>

    <head>

        <title>Inserted Text Example</title>

    </head>

    <body>

        <p>I want to drink <del>cola</del> <ins>wine</ins></p>

    </body>

</html>
```

This will produce the following result:



Deleted Text

Anything that appears within `...` element, is displayed as deleted text.

Example:

```
<!DOCTYPE html>

<html>

    <head>

        <title>Deleted Text Example</title>

    </head>

    <body>

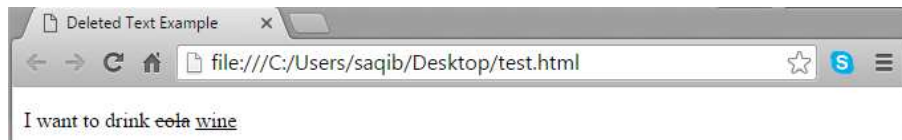
        <p>I want to drink <del>cola</del> <ins>wine</ins></p>

    </body>

</html>
```

This will produce the following result:





Larger Text

The content of the `<big>...</big>` element is displayed one font size larger than the rest of the text surrounding it as shown below:

```
<!DOCTYPE html>

<html>

  <head>

    <title>Larger Text Example</title>

  </head>

  <body>

    <p>The following word uses a <big>big</big> typeface.</p>

  </body>

</html>
```

This will produce the following result:



Smaller Text

The content of the `<small>...</small>` element is displayed one font size smaller than the rest of the text surrounding it as shown below:

```
<!DOCTYPE html>

<html>

    <head>

        <title>Smaller Text Example</title>

    </head>

    <body>

        <p>The following word uses a <small>small</small> typeface.</p>

    </body>

</html>
```

This will produce the following result:



Grouping Content

The `<div>` and `` elements allow you to group together several elements to create sections or subsections of a page.

For example, you might want to put all of the footnotes on a page within a `<div>` element to indicate that all of the elements within that `<div>` element relate to the footnotes. You might then attach a style to this `<div>` element so that they appear using a special set of style rules.




```
<!DOCTYPE html>

<html>

    <head>

        <title>Div Tag Example</title>

    </head>

    <body>

        <div id="menu" align="middle" >

            <a href="/index.htm">HOME</a> |

            <a href="/about/contact_us.htm">CONTACT</a> |

            <a href="/about/index.htm">ABOUT</a>

        </div>

        <div id="content" align="left" bgcolor="white">

            <h5>Content Articles</h5>

            <p>Actual content goes here.....</p>

        </div>

    </body>

</html>
```

This will produce the following result:



The `` element, on the other hand, can be used to group inline elements only. So, if you have a part of a sentence or paragraph which you want to group together, you could use the `` element as follows Example:



```
<!DOCTYPE html>

<html>

    <head>

        <title>Span Tag Example</title>

    </head>

    <body>

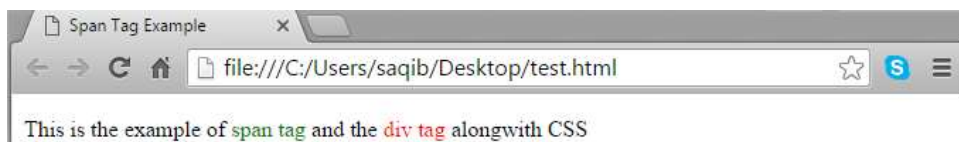
        <p>This is the example of <span style="color:green">span tag</span> and the

        <span style="color:red">div tag</span> alongwith CSS</p>

    </body>

</html>
```

This will produce the following result:



4.1.2.7 HTML Links

HTML Links-Hyperlinks

HTML links are hyperlinks. You can click on a link and jump to another document. When you move the mouse over a link, the mouse arrow will turn into a little hand.

Note: A link does not have to be text. It can be an image or any other HTML element.

HTML Links - Syntax

In HTML, links are defined with the `<a>` tag:

```
<a href="url">Link text</a>
```

Example



```
<a href="http://www.w3schools.com/html/">Visit our HTML
tutorial</a>
```

The **href** attribute specifies the destination address (<http://www.w3schools.com/html/>) of the link. The **link text** is the visible part (Visit our HTML tutorial). Clicking on the link text will send you to the specified address.

Note: Without a forward slash on subfolder addresses, you might generate two requests to the server. Many servers will automatically add a forward slash to the address, and then create a new request.

Local Links

The example above used an absolute URL (A full web address). A local link (link to the same web site) is specified with a relative URL (without <http://www....>).

Example

```
<a href="html_images.asp">HTML Images</a>
```

HTML Link Colors

- By default, a link will appear like this (in all browsers):
- An unvisited link is underlined and blue
- A visited link is underlined and purple
- An active link is underlined and red
- You can change the default colors, by using styles:

Example

```
<style>
a:link {color:green; background-color:transparent; text-decoration:none}
a:visited {color:blue; background-color:transparent; text-decoration:none}
a:hover {color:red; background-color:transparent; text-decoration:underline}
a:active {color:yellow; background-color:transparent; text-
decoration:underline}
</style>
```



HTML Links - The target Attribute

The **target** attribute specifies where to open the linked document.

The target attribute can have one of the following values:

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

This example will open the linked document in a new browser window/tab:

Example

```
<a href="http://www.w3schools.com/" target="_blank">Visit  
W3Schools!</a>
```

Tip: If your webpage is locked in a frame, you can use `target="_top"` to break out of the frame:

Example

```
<a href="http://www.w3schools.com/html/" target="_top">HTML5  
tutorial!</a>
```

HTML Links - Image as Link

It is common to use images as links:

```
<a href="default.asp">  
    
</a>
```

Example

HTML Links - Create a Bookmark

- HTML bookmarks are used to allow readers to jump to specific parts of a Web page.
- Bookmarks can be useful if your webpage is very long.
- To make a bookmark, you must first create the bookmark, and then add a link to it.



- When the link is clicked, the page will scroll to the location with the bookmark.

Example

```
<h2 id="tips">Useful Tips Section</h2>
```

First, create a bookmark with the id attribute:

```
<a href="#tips">Visit the Useful Tips Section</a>
```

Then, add a link to the bookmark ("Useful Tips Section"), from within the same page:

Or, add a link to the bookmark ("Useful Tips Section"), from another page:

```
<a href="html_tips.html#tips">Visit the Useful Tips Section</a>
```

Example

External Paths

External pages can be referenced with a full URL or with a path relative to the current web page. This example uses a full URL to link to a web page:

Example

```
<a href="http://www.w3schools.com/html/default.asp">HTML  
tutorial</a>
```

This example links to a page located in the html folder on the current web site:

Example

```
<a href="/html/default.asp">HTML tutorial</a>
```

This example links to a page located in the same folder as the current page:

Example

```
<a href="default.asp">HTML tutorial</a>
```



Chapter Summary

- Use the **<a>** element to define a link
- Use the **href** attribute to define the link address
- Use the **target** attribute to define where to open the linked document
- Use the **** element (inside **<a>**) to use an image as a link
- Use the **id** attribute (**id="value"**) to define bookmarks in a page
- Use the **href** attribute (**href="#value"**) to link to the bookmark

4.1.2.8 HTML Head

The HTML **<head>** Element

- The **<head>** element is a container for metadata (data about data) and is placed between the **<html>** tag and the **<body>** tag.
- HTML metadata is data about the HTML document. Metadata is not displayed.
- Metadata typically define the document title, character set, styles, links, scripts, and other meta information.
- The following tags describe metadata: **<title>**, **<style>**, **<meta>**, **<link>**, **<script>**, and **<base>**.

The HTML **<title>** Element

- The **<title>** element defines the title of the document, and is required in all HTML/XHTML documents.
- The **<title>** element defines a title in the browser tab
- provides a title for the page when it is added to favorites
- displays a title for the page in search engine results
- A simple HTML document:

Example

```
<!DOCTYPE html>
<html>

<head>
  <title>Page Title</title>
</head>

<body>
The content of the document.....
</body>

</html>
```



The HTML <style> Element

The <style> element is used to define style information for a single HTML page:

Example

```
<style>
  body {background-color: powderblue;}
  h1 {color: red;}
  p {color: blue;}
</style>
```

The HTML <link> Element

The <link> element is used to link to external style sheets:

Example

```
<link rel="stylesheet" href="mystyle.css">
```

The HTML <meta> Element

- The <meta> element is used to specify which character set is used, page description, keywords, author, and other metadata.
- Metadata is used by browsers (how to display content), by search engines (keywords), and other web services.

Define the character set used:

```
<meta charset="UTF-8">
```

Define a description of your web page:

```
<meta name="description" content="Free Web tutorials">
```

Define keywords for search engines:

```
<meta name="keywords" content="HTML, CSS, XML, JavaScript">
```

Define the author of a page:

```
<meta name="author" content="Hege Refsnes">
```



Refresh document every 30 seconds:

Example of <meta> tags:

```
<meta http-equiv="refresh" content="30">
```

```
<meta charset="UTF-8">  
<meta name="description" content="Free Web tutorials">  
<meta name="keywords" content="HTML,CSS,XML,JavaScript">  
<meta name="author" content="Hege Refsnes">
```

Example

Setting the Viewport

HTML5 introduced a method to let web designers take control over the viewport, through the <meta> tag. The viewport is the user's visible area of a web page. It varies with the device, and will be smaller on a mobile phone than on a computer screen.

You should include the following <meta> viewport element in all your web pages:

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

A <meta> viewport element gives the browser instructions on how to control the page's dimensions and scaling. The width=device-width part sets the width of the page to follow the screen-width of the device (which will vary depending on the device). The initial-scale=1.0 part sets the initial zoom level when the page is first loaded by the browser.

Here is an example of a web page *without* the viewport meta tag, and the same web page *with* the viewport meta tag:

Tip: If you are browsing this page with a phone or a tablet, you can click on the two links below to see the difference.

Without the viewport meta tag





With the viewport meta tag



The HTML <script> Element

The <script> element is used to define client-side JavaScript. This JavaScript writes "Hello JavaScript!" into an HTML element with id="demo":

Example

```
<script>
function myFunction {
    document.getElementById("demo").innerHTML = "Hello JavaScript!";
}
</script>
```

Omitting <html>, <head> and <body>?



According to the HTML5 standard; the <html>, the <body>, and the <head> tag can be omitted. The following code will validate as HTML5:

Example

```
<!DOCTYPE html>
<title>Page Title</title>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>
```

Note: W3Schools does not recommend omitting the <html> and <body> tags. Omitting these tags can crash DOM or XML software and produce errors in older browsers (IE9).However, omitting the <head> tag has been a common practice for quite some time now.

HTML head Elements

Tag	Description
<head>	Defines information about the document
<title>	Defines the title of a document
<link>	Defines the relationship between a document and an external source
<meta>	Defines metadata about an HTML document
<script>	Defines a client-side script
<style>	Defines style information for a document

4.1.2.9 HTML Images

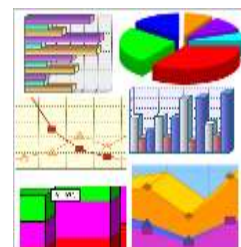
JPG Images



GIF Images



PNG Images



Example

```
<!DOCTYPE html>
<html>
<body>

<h2>Spectacular Mountain</h2>


</body>
</html>
```

HTML Images Syntax

In HTML, images are defined with the **** tag. The **** 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:

```

```

The alt Attribute

The alt attribute provides an alternate text for an image, if the user for some reason cannot view it (because of slow connection, an error in the src attribute, or if the user uses a screen reader).

If a browser cannot find an image, it will display the value of the alt attribute: Example

```

```

The alt attribute is required. A web page will not validate correctly without it.

HTML Screen Readers

A screen reader is a software program that reads the HTML code, converts the text, and allows the user to "listen" to the content. Screen readers are useful for people who are blind, visually impaired, or learning disabled.

Image Size - Width and Height

You can use the **style** attribute to specify the width and height of an image.



The values are specified in pixels (use px after the value): Example

```

```

Alternatively, you can use the **width** and **height** attributes. Here, the values are specified in pixels by default:

Example

```

```

Note: Always specify the width and height of an image. If width and height are not specified, the page will flicker while the image loads.

Width and Height, or Style?

Both the width, height, and style attributes are valid in HTML5. However, we suggest using the style attribute. It prevents internal or external styles sheets from changing the original size of images:

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
img {
    width:100%;
}
</style>
</head>
<body>




</body>
```

Images in another Folder

If not specified, the browser expects to find the image in the same folder as the web page. However, it is common to store images in a sub-folder. You must then include the folder name in the src attribute:



Example

```

```

Images on another Server

Some web sites store their images on image servers. Actually, you can access images from any web address in the world:

Example

```

```

Animated Images

The GIF standard allows animated images:

Example

```

```

Note that the syntax of inserting animated images is no different from non-animated images.

Using an Image as a Link

To use an image as a link, simply nest the tag inside the <a> tag:

Example

```
<a href="default.asp">  
    
</a>
```

Note: border:0; is added to prevent IE9 (and earlier) from displaying a border around the image (when the image is a link).

Image Floating



Use the CSS **float** property to let the image float to the right or to the left of a text:

Chapter Summary

```
<p>
The image will float to the right of the text.</p>

<p>
The image will float to the left of the text.</p>
```

- Use the HTML **** element to define an image
- Use the HTML **src** attribute to define the URL of the image
- Use the HTML **alt** attribute to define an alternate text for an image, if it cannot be displayed
- Use the HTML **width** and **height** attributes to define the size of the image
- Use the CSS **width** and **height** properties to define the size of the image (alternatively)
- Use the CSS **float** property to let the image float

Note: Loading images takes time. Large images can slow down your page. Use images carefully.

4.1.2.10 HTML Tables

HTML Table Example

HTML Table							
Cell 1	Cell 2						
	<table><tr><th colspan="2">Nested HTML Table</th></tr><tr><td>Cell 2.1</td><td>Cell 2.2</td></tr><tr><td>Cell 2.3</td><td>Cell 2.4</td></tr></table>	Nested HTML Table		Cell 2.1	Cell 2.2	Cell 2.3	Cell 2.4
	Nested HTML Table						
	Cell 2.1	Cell 2.2					
Cell 2.3	Cell 2.4						

Defining an HTML Table

An HTML table is defined with the **<table>** tag. Each table row is defined with the **<tr>** tag. A table header is defined with the **<th>** tag. By default, table headings are bold and centered. A table data/cell is defined with the **<td>** tag.



Example

```
<table style="width:100%">
  <tr>
    <th>Firstname</th>
    <th>Lastname</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
  </tr>
</table>
```

Note: The `<td>` elements are the data containers of the table. They can contain all sorts of HTML elements; text, images, lists, other tables, etc.

HTML Table - Adding a Border

If you do not specify a border for the table, it will be displayed without borders. A border is set using the CSS **border** property:

Example

```
table, th, td {
  border: 1px solid black;
}
```

Remember to define borders for both the table and the table cells.

HTML Table - Collapsed Borders

If you want the borders to collapse into one border, add the CSS **border-collapse** property:

Example

```
table, th, td {
  border: 1px solid black;
  border-collapse: collapse;
}
```

HTML Table - Adding Cell Padding



Cell padding specifies the space between the cell content and its borders. If you do not specify a padding, the table cells will be displayed without padding.

To set the padding, use the CSS **padding** property:

Example

```
th, td {  
    padding: 15px;  
}
```

HTML Table - Left-align Headings

By default, table headings are bold and centered. To left-align the table headings, use the CSS **text-align** property:

Example

```
th {  
    text-align: left;  
}
```

HTML Table - Adding Border Spacing

Border spacing specifies the space between the cells. To set the border spacing for a table, use the CSS **border-spacing** property:

Example

```
table {  
    border-spacing: 5px;  
}
```

Note: If the table has collapsed borders, border-spacing has no effect.

HTML Table - Cells that Span Many Columns

To make a cell span more than one column, use the **colspan** attribute:

Example

```
<table style="width:100%">  
<tr>  
    <th>Name</th>  
    <th colspan="2">Telephone</th>
```



```
</tr>
<tr>
  <td>Bill Gates</td>
  <td>55577854</td>
  <td>55577855</td>
</tr>
</table>
```

HTML Table - Cells that Span Many Rows

To make a cell span more than one row, use the **rowspan** attribute:

Example

```
<table style="width:100%">
  <tr>
    <th>Name:</th>
    <td>Bill Gates</td>
  </tr>
  <tr>
    <th rowspan="2">Telephone:</th>
    <td>55577854</td>
  </tr>
  <tr>
    <td>55577855</td>
  </tr>
</table>
```

HTML Table - Adding a Caption

To add a caption to a table, use the **<caption>** tag:

Example

```
<table style="width:100%">
  <caption>Monthly savings</caption>
  <tr>
    <th>Month</th>
    <th>Savings</th>
  </tr>
  <tr>
    <td>January</td>
    <td>$100</td>
  </tr>
</table>
```



```

</tr>
<tr>
  <td>February</td>
  <td>$50</td>
</tr>
</table>

```

Note: The <caption> tag must be inserted immediately after the <table> tag.

A Special Style for One Table

To define a special style for a special table, add an **id** attribute to the table:

Example

```

<table id="t01">
  <tr>
    <th>Firstname</th>
    <th>Lastname</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
  </tr>
</table>

```

Now you can define a special style for this table:

```

table#t01 {
  width: 100%;
  background-color: #f1f1c1;
}

```

And add more styles:

```

table#t01 tr:nth-child(even) {
  background-color: #eee;
}
table#t01 tr:nth-child(odd) {
  background-color: #fff;
}
table#t01 th {
  color: white;
  background-color: black;
}

```



```
}
```

Chapter Summary

- Use the HTML **<table>** element to define a table
- Use the HTML **<tr>** element to define a table row
- Use the HTML **<td>** element to define a table data
- Use the HTML **<th>** element to define a table heading
- Use the HTML **<caption>** element to define a table caption
- Use the CSS **border** property to define a border
- Use the CSS **border-collapse** property to collapse cell borders
- Use the CSS **padding** property to add padding to cells
- Use the CSS **text-align** property to align cell text
- Use the CSS **border-spacing** property to set the spacing between cells
- Use the **colspan** attribute to make a cell span many columns
- Use the **rowspan** attribute to make a cell span many rows
- Use the **id** attribute to uniquely define one table

4.1.2.11 HTML Lists

HTML List Example

An Unordered List:

- Item
- Item
- Item
- Item

An Ordered List:

1. First item
2. Second item
3. Third item
4. Fourth item

Unordered HTML List

An unordered list starts with the **** tag. Each list item starts with the **** tag.

The list items will be marked with bullets (small black circles) by default:

Example



```
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

Unordered HTML List - Choose List Item Marker

The CSS **list-style-type** property is used to define the style of the list item marker:

Value	Description
disc	Sets the list item marker to a bullet (default)
circle	Sets the list item marker to a circle
square	Sets the list item marker to a square
none	The list items will not be marked

Example - Disc

```
<ul style="list-style-type:disc">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

Example - Circle

```
<ul style="list-style-type:circle">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

Example - Square

```
<ul style="list-style-type:square">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

Example - None



```
<ul style="list-style-type:none">
<li>Coffee</li>
<li>Tea</li>
<li>Milk</li>
</ul>
```

Ordered HTML List

An ordered list starts with the **** tag. Each list item starts with the **** tag.

The list items will be marked with numbers by default:

Example

```
<ol>
<li>Coffee</li>
<li>Tea</li>
<li>Milk</li>
</ol>
```

Ordered HTML List - The Type Attribute

The **type** attribute of the **** tag, defines the type of the list item marker:

Type	Description
type="1"	The list items will be numbered with numbers (default)
Type="A"	The list items will be numbered with uppercase letters
Type="a"	The list items will be numbered with lowercase letters
Type="I"	The list items will be numbered with uppercase roman numbers
Type="i"	The list items will be numbered with lowercase roman numbers

Numbers:

```
<ol type="1">
```



```
<li>Coffee</li>
<li>Tea</li>
<li>Milk</li>
</ol>
```

Uppercase Letters:

```
<ol type="A">
<li>Coffee</li>
<li>Tea</li>
<li>Milk</li>
</ol>
```

Lowercase Letters:

```
<ol type="a">
<li>Coffee</li>
<li>Tea</li>
<li>Milk</li>
</ol>
```

Uppercase Roman Numbers:

```
<ol type="I">
<li>Coffee</li>
<li>Tea</li>
<li>Milk</li>
</ol>
```

Lowercase Roman Numbers:

```
<ol type="i">
<li>Coffee</li>
<li>Tea</li>
<li>Milk</li>
</ol>
```

HTML Description Lists

HTML also supports description lists. A description list is a list of terms, with a description of each term. The **<dl>** tag defines the description list, the **<dt>** tag defines the term (name), and the **<dd>** tag describes each term:

Example



```
<dl>
  <dt>Coffee</dt>
  <dd>- black hot drink</dd>
  <dt>Milk</dt>
  <dd>- white cold drink</dd>
</dl>
```

Nested HTML Lists

List can be nested (lists inside lists):

Example

```
<ul>
  <li>Coffee</li>
  <li>Tea
    <ul>
      <li>Black tea</li>
      <li>Green tea</li>
    </ul>
  </li>
  <li>Milk</li>
</ul>
```

Note: List items can contain new list, and other HTML elements, like images and links, etc.

Horizontal Lists

HTML lists can be styled in many different ways with CSS. One popular way is to style a list horizontally, to create a menu:

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  overflow: hidden;
  background-color: #333333;
}
```



```
li {  
    float: left;  
}  
  
li a {  
    display: block;  
    color: white;  
    text-align: center;  
    padding: 16px;  
    text-decoration: none;  
}  
  
li a:hover {  
    background-color: #111111;  
}  
</style>  
</head>  
<body>  
  
<ul>  
    <li><a href="#home">Home</a></li>  
    <li><a href="#news">News</a></li>  
    <li><a href="#contact">Contact</a></li>  
    <li><a href="#about">About</a></li>  
</ul>  
  
</body>  
</html>
```

Chapter Summary

- Use the HTML **** element to define an unordered list
- Use the CSS **list-style-type** property to define the list item marker
- Use the HTML **** element to define an ordered list
- Use the HTML **type** attribute to define the numbering type
- Use the HTML **** element to define a list item
- Use the HTML **<dl>** element to define a description list
- Use the HTML **<dt>** element to define the description term
- Use the HTML **<dd>** element to describe the term in a description list
- Lists can be nested inside lists
- List items can contain other HTML elements



- Use the CSS property **float:left** or **display:inline** to display a list horizontally

4.1.2.12 Html Block and inline elements

Every HTML element has a default display value depending on what type of element it is. The default display value for most elements is block or inline.

Block-level Elements

A block-level element always starts on a new line and takes up the full width available (stretches out to the left and right as far as it can). The `<div>` element is a block-level element.

Examples of block-level elements:

- `<div>`
- `<h1>` - `<h6>`
- `<p>`
- `<form>`

Inline Elements

An inline element does not start on a new line and only takes up as much width as necessary.

This is an inline `` element inside a paragraph.

Examples of inline elements:

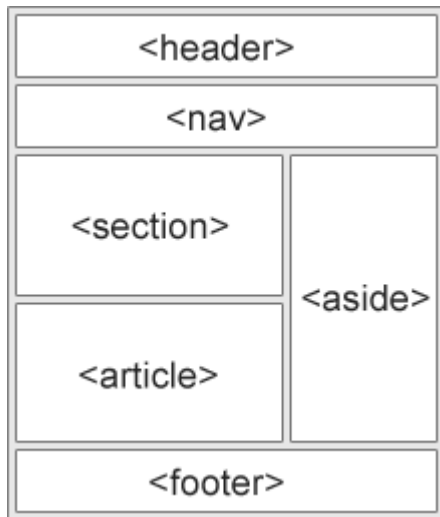
- ``
- `<a>`
- ``

4.1.2.13 HTML Layouts

HTML Layout Elements

Websites often display content in multiple columns (like a magazine or newspaper).

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



- <header> - Defines a header for a document or a section
- <nav> - Defines a container for navigation links
- <section> - Defines a section in a document
- <article> - Defines an independent self-contained article
- <aside> - Defines content aside from the content (like a sidebar)
- <footer> - Defines a footer for a document or a section
- <details> - Defines additional details
- <summary> - Defines a heading for the <details> element

HTML Layout Techniques

There are four different ways to create multicolumn layouts. Each way has its pros and cons:

- HTML tables
- CSS float property
- CSS framework
- CSS flexbox

Which One to Choose?

HTML Tables

The <table> element was not designed to be a layout tool! The purpose of the <table> element is to display tabular data. So, do not use tables for your page layout! They will bring a mess into your code. And imagine how hard it will be to redesign your site after a couple of months.

Tip: Do NOT use tables for your page layout!

CSS Frameworks

If you want to create your layout fast, you can use a framework, like [W3.CSS](#) or [Bootstrap](#).

CSS Floats

It is common to do entire web layouts using the CSS float property. Float is easy to learn - you just need to remember how the float and clear properties work. Disadvantages: Floating elements are tied to the document flow, which may harm the flexibility.



4.1.2.14 HTML Forms

HTML Forms are required, when you want to collect some data from the site visitor. For example, during user registration you would like to collect information such as name, email address, credit card, etc.

A form will take input from the site visitor and then will post it to a back-end application such as CGI, ASP Script or PHP script etc. The back-end application will perform required processing on the passed data based on defined business logic inside the application.

There are various form elements available like text fields, textarea fields, drop-down menus, radio buttons, checkboxes, etc.

The HTML <form> tag is used to create an HTML form and it has following syntax:

```
<form action="Script URL" method="GET|POST">
```

form elements like input, textarea etc.

```
</form>
```

Form Attributes

Apart from common attributes, following is a list of the most frequently used form attributes:

Attribute	Description
action	Backend script ready to process your passed data.
method	Method to be used to upload data. The most frequently used are GET and POST methods.
target	Specify the target window or frame where the result of the script will be displayed. It takes values like _blank, _self, _parent etc.
enctype	You can use the enctype attribute to specify how the browser encodes the data before it sends it to the server. Possible values are:
application/x-www-form-urlencoded - This is the standard method most forms use in simple scenarios.	multipart/form-data - This is used when you want to upload binary data in the form of files like image, word file etc.

HTML Form Controls

There are different types of form controls that you can use to collect data using HTML form:



- Text Input Controls
- Checkboxes Controls
- Radio Box Controls
- Select Box Controls
- File Select boxes
- Hidden Controls
- Clickable Buttons
- Submit and Reset Button

Text Input Controls

There are three types of text input used on forms:

- Single-line text input controls - This control is used for items that require only one line of user input, such as search boxes or names. They are created using HTML `<input>` tag.
- Password input controls - This is also a single-line text input but it masks the character as soon as a user enters it. They are also created using HTML `<input>` tag.
- Multi-line text input controls - This is used when the user is required to give details that may be longer than a single sentence. Multi-line input controls are created using HTML `<textarea>` tag.

Single-line text input controls

This control is used for items that require only one line of user input, such as search boxes or names. They are created using HTML `<input>` tag.

Example

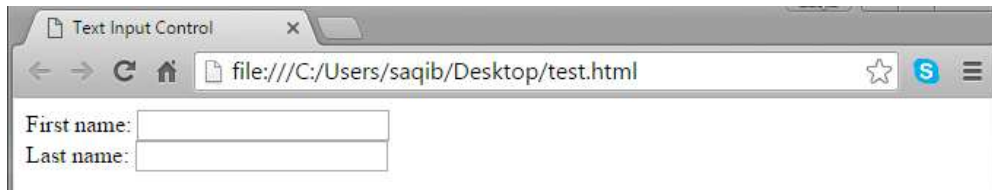
Here is a basic example of a single-line text input used to take first name and last name:

```
<!DOCTYPE html>
<html>
<head>
  <title>Text Input Control</title>
</head>
<body>
  <form>
    First name:
    <input type="text" name="first_name" />
    <br>
    Last name:
    <input type="text" name="last_name" />
  </form>
</body>
```



```
</html>
```

This will produce the following result:



Attributes

Following is the list of attributes for `<input>` tag for creating text field.

Attribute	Description
type	Indicates the type of input control and for text input control it will be set to text.
name	Used to give a name to the control which is sent to the server to be recognized and get the value.
value	This can be used to provide an initial value inside the control.
size	Allows to specify the width of the text-input control in terms of characters.
maxlength	Allows to specify the maximum number of characters a user can enter into the text box.

Password Input controls

This is also a single-line text input but it masks the character as soon as a user enters it. They are also created using HTML `<input>` tag but type attribute is set to password.

Example

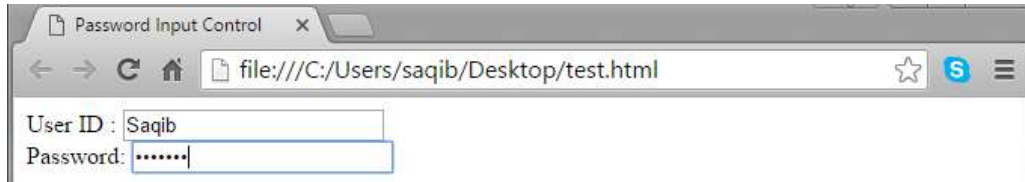
Here is a basic example of a single-line password input used to take user password:

```
<!DOCTYPE html>
<html>
<head>
  <title>Password Input Control</title>
</head>
<body>
  <form>
    User ID :
    <input type="text" name="user_id" />
    <br/>
    Password:
    <input type="password" name="password" />
```



```
</form>
</body>
</html>
```

This will produce the following result:



Multiple-Line Text Input Controls

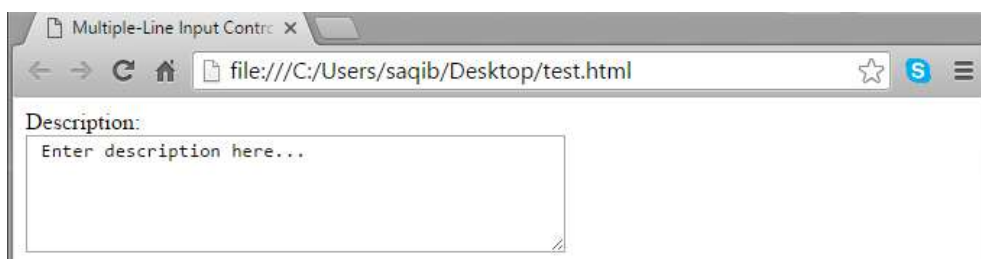
This is used when the user is required to give details that may be longer than a single sentence. Multi-line input controls are created using HTML `<textarea>` tag.

Example

Here is a basic example of a multi-line text input used to take item description:

```
<!DOCTYPE html>
<html>
<head>
  <title>Multiple-Line Input Control</title>
</head>
<body>
  <form>
    Description:
    <br />
    <textarea rows="5" cols="50" name="description"> Enter description here...
  </textarea>
  </form>
</body>
</html>
```

This will produce the following result:



Attributes

Following is the list of attributes for `<textarea>` tag.



Attribute	Description
name	Used to give a name to the control which is sent to the server to be recognized and get the value.
rows	Indicates the number of rows of text area box.
cols	Indicates the number of columns of text area box

Checkbox Control

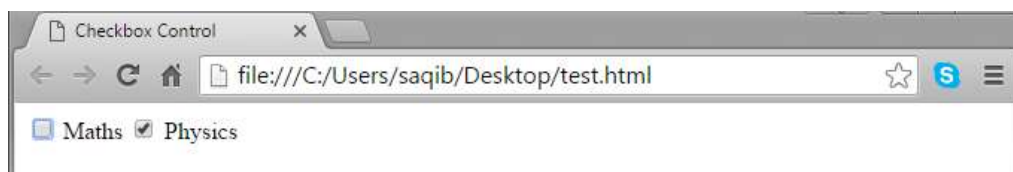
Checkboxes are used when more than one option is required to be selected. They are also created using HTML `<input>` tag but type attribute is set to checkbox.

Example

Here is an example HTML code for a form with two checkboxes:

```
<!DOCTYPE html>
<html>
<head>
  <title>Checkbox Control</title>
</head>
<body>
  <form>
    <input type="checkbox" name="maths" value="on">
    Maths
    <input type="checkbox" name="physics" value="on">
    Physics
  </form>
</body>
</html>
```

This will produce the following result:



Attributes	Following is the list of attributes for <code><checkbox></code> tag.
Attribute	Description
type	Indicates the type of input control and for checkbox input control it will be set to checkbox.
name	Used to give a name to the control which is sent to the server to be recognized and get the value.



value	The value that will be used if the checkbox is selected.
checked	Set to checked if you want to select it by default.

Radio Button Control

Radio buttons are used when out of many options, just one option is required to be selected. They are also created using HTML `<input>` tag but type attribute is set to radio.

Example

Here is example HTML code for a form with two radio buttons:

```
<!DOCTYPE html>
<html>
<head>
  <title>Radio Box Control</title>
</head>
<body>
  <form>
    <input type="radio" name="subject" value="maths">
    Maths
    <input type="radio" name="subject" value="physics">
    Physics
  </form>
</body>
</html>
```

This will produce the following result:



Attributes

Following is the list of attributes for radio button.

Attribute	Description
type	Indicates the type of input control and for checkbox input control it will be set to radio.
name	Used to give a name to the control which is sent to the server to be recognized and get the value.



value	The value that will be used if the radio box is selected.
checked	Set to checked if you want to select it by default.

Select Box Control

A select box, also called drop down box which provides option to list down various options in the form of drop down list, from where a user can select one or more options.

Example

Here is example HTML code for a form with one drop down box

```
<!DOCTYPE html>
<html>
<head>
  <title>Select Box Control</title>
</head>
<body>
  <form>
    <select name="dropdown">
      <option value="Maths" selected="selected">Maths</option>
      <option value="Physics">Physics</option>
    </select>
  </form>
</body>
</html>
```

This will produce the following result:



Attributes

Following is the list of important attributes of <select> tag:

Attribute	Description
name	Used to give a name to the control which is sent to the server to be recognized and get the value.



size	This can be used to present a scrolling list box.
multiple	If set to "multiple" then allows a user to select multiple items from the menu.

Following is the list of important attributes of <option> tag:

Attribute	Description
value	The value that will be used if an option in the select box box is selected.
selected	Specifies that this option should be the initially selected value when the page loads.
label	An alternative way of labeling options

File Upload Box

If you want to allow a user to upload a file to your web site, you will need to use a file upload box, also known as a file select box. This is also created using the <input> element but type attribute is set to file.

Example

Here is example HTML code for a form with one file upload box:

```
<!DOCTYPE html>
<html>
<head>
  <title>File Upload Box</title>
</head>
<body>
  <form>
    <input type="file" name="fileupload" accept="image/*" />
  </form>
</body>
</html>
```

This will produce the following result:



Attributes

Following is the list of important attributes of file upload box:

Attribute	Description
name	Used to give a name to the control which is sent to the server to be recognized and get the value.
accept	Specifies the types of files that the server accepts.

Button Controls

There are various ways in HTML to create clickable buttons. You can also create a clickable button using `<input>` tag by setting its type attribute to button. The type attribute can take the following values:

Type	Description
submit	This creates a button that automatically submits a form.
reset	This creates a button that automatically resets form controls to their initial values.
button	This creates a button that is used to trigger a client-side script when the user clicks that button.
image	This creates a clickable button but we can use an image as background of the button.

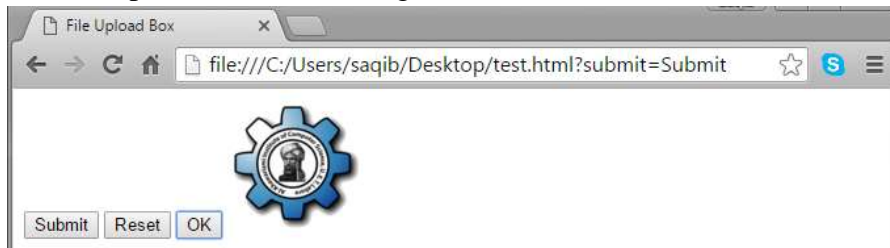
Example

Here is example HTML code for a form with three types of buttons:

```
<!DOCTYPE html>
<html>
<head>
  <title>File Upload Box</title>
</head>
<body>
  <form>
    <input type="submit" name="submit" value="Submit" />
    <input type="reset" name="reset" value="Reset" />
    <input type="button" name="ok" value="OK" />
    <input type="image" name="imagebutton" src="logo.png" width="100"
height="100"/>
  </form>
</body>
</html>
```



This will produce the following result:



Hidden Form Controls

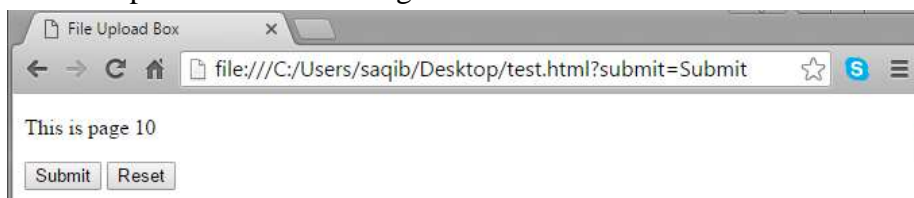
Hidden form controls are used to hide data inside the page which later on can be pushed to the server. This control hides inside the code and does not appear on the actual page. For example, following hidden form is being used to keep current page number. When a user will click next page then the value of hidden control will be sent to the web server and there it will decide which page will be displayed next based on the passed current page.

Example

Here is example HTML code to show the usage of hidden control:

```
<!DOCTYPE html>
<html>
<head>
  <title>File Upload Box</title>
</head>
<body>
  <form>
    <p>This is page 10</p>
    <input type="hidden" name="pagename" value="10" />
    <input type="submit" name="submit" value="Submit" />
    <input type="reset" name="reset" value="Reset" />
  </form>
</body>
</html>
```

This will produce the following result:



4.1.2.15 HTML Iframes

An iframe is used to display a web page within a web page.



Iframe Syntax

An HTML iframe is defined with the **<iframe>** tag:

```
<iframe src="URL"></iframe>
```

The **src** attribute specifies the URL (web address) of the inline frame page.

Iframe - Set Height and Width

Use the **height** and **width** attributes to specify the size of the iframe. The attribute values are specified in pixels by default, but they can also be in percent (like "80%").

Example

```
<iframe src="demo_iframe.htm" height="200" width="300"></iframe>
```

Iframe - Remove the Border

By default, an iframe has a border around it. To remove the border, add the **style** attribute and use the CSS **border** property:

Example

```
<iframe src="demo_iframe.htm" style="border:none;"></iframe>
```

With CSS, you can also change the size, style and color of the iframe's border:

Example

```
<iframe src="demo_iframe.htm" style="border:2px solid grey;"></iframe>
```

Iframe - Target for a Link

An iframe can be used as the target frame for a link. The **target** attribute of the link must refer to the **name** attribute of the iframe:

Example

```
<iframe src="demo_iframe.htm" name="iframe_a"></iframe>  
<p><a href="http://www.w3schools.com" target="iframe_a">W3Schools.com</a></p>
```



HTML iframe Tag

Tag	Description
<iframe>	Defines an inline frame

4.1.2.16 HTML Colours

Colors are very important to give a good look and feel to your website. You can specify colors on page level using <body> tag or you can set colors for individual tags using bgcolor attribute.

The <body> tag has following attributes which can be used to set different colors:

- bgcolor - sets a color for the background of the page.
- text - sets a color for the body text.
- alink - sets a color for active links or selected links.
- link - sets a color for linked text.
- vlink - sets a color for visited links - that is, for linked text that you have already clicked on.

4.1.2.17 HTML INPUT Types

This chapter describes the different input types for the <input> element.

Input Type Text

<input type="text"> defines a **one-line text input field**:

Example

```
<form>
First name:<br>
<input type="text" name="firstname"><br>
Last name:<br>
<input type="text" name="lastname">
</form>
```

This is how the HTML code above will be displayed in a browser:

First name:

Last name:



Input Type Password

`<input type="password">` defines a **password field**:

Example

```
<form>
User name:<br>
<input type="text" name="username"><br>
User password:<br>
<input type="password" name="psw">
</form>
```

This is how the HTML code above will be displayed in a browser:

User name:

User password:

The characters in a password field are masked (shown as asterisks or circles).

Input Type Submit

- `<input type="submit">` defines a button for **submitting** form data to a **form-handler**.
- The form-handler is typically a server page with a script for processing input data.
- The form-handler is specified in the form's **action** attribute:

Example

```
<form action="action_page.php">
First name:<br>
<input type="text" name="firstname" value="Mickey"><br>
Last name:<br>
<input type="text" name="lastname" value="Mouse"><br><br>
<input type="submit" value="Submit">
</form>
```

This is how the HTML code above will be displayed in a browser:

First name:

Last name:



If you omit the submit button's value attribute, the button will get a default text:

Example

```
<form action="action_page.php">  
  First name:<br>  
  <input type="text" name="firstname" value="Mickey"><br>  
  Last name:<br>  
  <input type="text" name="lastname" value="Mouse"><br><br>  
  <input type="submit">  
</form>
```

Input Type Reset

<input type="reset"> defines a **reset button** that will reset all form values to their default values:

Example

```
<form action="action_page.php">  
  First name:<br>  
  <input type="text" name="firstname" value="Mickey"><br>  
  Last name:<br>  
  <input type="text" name="lastname" value="Mouse"><br><br>  
  <input type="submit" value="Submit">  
  <input type="reset">  
</form>
```

This is how the HTML code above will be displayed in a browser:

First name:

Last name:

If you change the input values and then click the "Reset" button, the form-data will be reset to the default values.



Input Type Radio

`<input type="radio">` defines a **radio button**.

Radio buttons let a user select ONLY ONE of a limited number of choices:

Example

```
<form>
<input type="radio" name="gender" value="male" checked> Male<br>
<input type="radio" name="gender" value="female"> Female<br>
<input type="radio" name="gender" value="other"> Other
</form>
```

This is how the HTML code above will be displayed in a browser:

☒ Male
☐ Female
☐ Other

Input Type Checkbox

`<input type="checkbox">` defines a **checkbox**.

Checkboxes let a user select ZERO or MORE options of a limited number of choices.

Example

```
<form>
<input type="checkbox" name="vehicle1" value="Bike"> I have a bike<br>
<input type="checkbox" name="vehicle2" value="Car"> I have a car
</form>
```

This is how the HTML code above will be displayed in a browser:

☐ I have a bike
☐ I have a car

Input Type Button

`<input type="button">` defines a **button**:

Example

```
<input type="button" onclick="alert('Hello World!')" value="Click Me!">
```



HTML5 Input Types

HTML5 added several new input types:

- color
- date
- datetime
- datetime-local
- email
- month
- number
- range
- search
- tel
- time
- url
- week

New input types that are not supported by older web browsers will behave as `<input type="text">`.

Input Type Number

- The `<input type="number">` defines a **numeric** input field.
- You can also set restrictions on what numbers are accepted.
- The following example displays a numeric input field, where you can enter a value from 1 to 5:

Example

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

The following example displays a numeric input field, where you can enter a value from 0 to 100, in steps of 10. The default value is 30:

Example

```
<form>
Quantity:
<input type="number" name="points" min="0" max="100" step="10" value="30">
```



```
</form>
```

Input Type Date

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

Example

```
<form>
  Birthday:
  <input type="date" name="bday">
</form>
```

You can also add restrictions to dates:

Example

```
<form>
  Enter a date before 1980-01-01:
  <input type="date" name="bday" max="1979-12-31"><br>
  Enter a date after 2000-01-01:
  <input type="date" name="bday" min="2000-01-02"><br>
</form>
```

Input Type Color

The `<input type="color">` is used for input fields that should contain a color.

Depending on browser support, a color picker can show up in the input field.

Example

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

Input Type Range

The `<input type="range">` is used for input fields that should contain a value within a range. Depending on browser support, the input field can be displayed as a slider control.

Example

```
<form>
  <input type="range" name="points" min="0" max="10">
</form>
```



You can use the following attributes to specify restrictions: min, max, step, value.

Input Type Month

The `<input type="month">` allows the user to select a month and year. Depending on browser support, a date picker can show up in the input field.

Example

```
<form>
  Birthday (month and year):
  <input type="month" name="bdaymonth">
</form>
```

Input Type Week

The `<input type="week">` allows the user to select a week and year. Depending on browser support, a date picker can show up in the input field.

Example

```
<form>
  Select a week:
  <input type="week" name="week_year">
</form>
```

Input Type Time

The `<input type="time">` allows the user to select a time (no time zone).

Depending on browser support, a time picker can show up in the input field.

Example

```
<form>
  Select a time:
  <input type="time" name="usr_time">
</form>
```

Input Type Datetime-local

The `<input type="datetime-local">` specifies a date and time input field, with no time zone. Depending on browser support, a date picker can show up in the input field.

Example

```
<form>
  Birthday (date and time):
```



```
<input type="datetime-local" name="bdaytime">  
</form>
```

Input Type Email

- The **<input type="email">** is used for input fields that should contain an e-mail address.
- Depending on browser support, the e-mail address can be automatically validated when submitted.
- Some smartphones recognize the email type, and adds ".com" to the keyboard to match email input.

Example

```
<form>  
E-mail:  
<input type="email" name="email">  
</form>
```

Input Type Search

The **<input type="search">** is used for search fields (a search field behaves like a regular text field).

Example

```
<form>  
Search Google:  
<input type="search" name="googlesearch">  
</form>
```

Input Type Tel

The **<input type="tel">** is used for input fields that should contain a telephone number. The tel type is currently supported only in Safari 8.

Example

```
<form>  
Telephone:  
<input type="tel" name="usrtel">  
</form>
```

Input Type Url

- The **<input type="url">** is used for input fields that should contain a URL address.



- Depending on browser support, the url field can be automatically validated when submitted.
- Some smartphones recognize the url type, and adds ".com" to the keyboard to match url input.

Example

```
<form>
Add your homepage:
<input type="url" name="homepage">
</form>
```

Input Restrictions

Here is a list of some common input restrictions (some are new in HTML5):

Attribute	Description-
disabled	Specifies that an input field should be disabled
max	Specifies the maximum value for an input field
maxlength	Specifies the maximum number of character for an input field
min	Specifies the minimum value for an input field
pattern	Specifies a regular expression to check the input value against
readonly	Specifies that an input field is read only (cannot be changed)
required	Specifies that an input field is required (must be filled out)
size	Specifies the width (in characters) of an input field
step	Specifies the legal number intervals for an input field
value	Specifies the default value for an input field



4.1.2.18 HTML Media

- Multimedia comes in many different formats. It can be almost anything you can hear or see.
- Examples: Pictures, music, sound, videos, records, films, animations, and more.
- Web pages often contain multimedia elements of different types and formats.
- In this chapter you will learn about the different multimedia formats.

Multimedia Formats

Multimedia elements (like sounds or videos) are stored in media files.

The most common way to discover the type of a file, is to look at the file extension. When a browser sees the file extension .htm or .html, it will treat the file as an HTML file. The .xml extension indicates an XML file, and the .css extension indicates a style sheet file. Pictures are recognized by extensions like .gif, .png and .jpg.

Multimedia files also have their own formats and different extensions like: .swf, .wav, .mp3, .mp4, .mpg, .wmv, and .avi.

Format	File	Description
MPEG	.mpg .mpeg	MPEG. Developed by the Moving Pictures Expert Group. The first popular video format on the web. Used to be supported by all browsers, but it is not supported in HTML5 (See MP4).
AVI	.avi	AVI (Audio Video Interleave). Developed by Microsoft. Commonly used in video cameras and TV hardware. Plays well on Windows computers, but not in web browsers.
WMV	.wmv	WMV (Windows Media Video). Developed by Microsoft. Commonly used in video cameras and TV hardware. Plays well on Windows computers, but not in web browsers.
QuickTime	.mov	QuickTime. Developed by Apple. Commonly used in video cameras and TV hardware. Plays well on Apple computers, but not in web browsers. (See MP4)
RealVideo	.rm .ram	RealVideo. Developed by Real Media to allow video streaming with low bandwidths. It is still used for online video and Internet TV, but does not play in web browsers.
Flash	.swf .flv	Flash. Developed by Macromedia. Often requires an extra component (plug-in) to play in web browsers.
Ogg	.ogg	Theora Ogg. Developed by the Xiph.Org Foundation. Supported by



		HTML5.
WebM	.webm	WebM. Developed by the web giants, Mozilla, Opera, Adobe, and Google. Supported by HTML5.
MPEG-4 or MP4	.mp4	MP4. Developed by the Moving Pictures Expert Group. Based on QuickTime. Commonly used in newer video cameras and TV hardware. Supported by all HTML5 browsers. Recommended by YouTube.

Sound Formats

MP3 is the newest format for compressed recorded music. The term MP3 has become synonymous with digital music.

If your website is about recorded music, MP3 is the choice.

Format	File	Description
MIDI	.mid .midi	MIDI (Musical Instrument Digital Interface). Main format for all electronic music devices like synthesizers and PC sound cards. MIDI files do not contain sound, but digital notes that can be played by electronics. Plays well on all computers and music hardware, but not in web browsers.
RealAudio	.rm .ram	RealAudio. Developed by Real Media to allow streaming of audio with low bandwidths. Does not play in web browsers.
WMA	.wma	WMA (Windows Media Audio). Developed by Microsoft. Commonly used in music players. Plays well on Windows computers, but not in web browsers.
AAC	.aac	AAC (Advanced Audio Coding). Developed by Apple as the default format for iTunes. Plays well on Apple computers, but not in web browsers.
WAV	.wav	WAV. Developed by IBM and Microsoft. Plays well on Windows, Macintosh, and Linux operating systems. Supported by HTML5.
Ogg	.ogg	Ogg. Developed by the Xiph.Org Foundation. Supported by HTML5.
MP3	.mp3	MP3 files are actually the sound part of MPEG files. MP3 is the most popular format for music players. Combines good compression (small files) with high quality. Supported by all browsers.
MP4	.mp4	MP4 is a video format, but can also be used for audio. MP4 video is the upcoming video format on the internet. This leads to automatic support for MP4 audio by all browsers.



Audio on the Web

- Before HTML5, there was no standard for playing audio files on a web page.
- Before HTML5, audio files could only be played with a plug-in (like flash).
- The HTML5 <audio> element specifies a standard way to embed audio in a web page.
- Example:

```
<audio controls>  
<source src="horse.ogg" type="audio/ogg">  
<source src="horse.mp3" type="audio/mpeg">  
</audio>
```

Playing Videos in HTML

- Before HTML5, there was no standard for showing videos on a web page.
- Before HTML5, videos could only be played with a plug-in (like flash).
- The HTML5 <video> element specifies a standard way to embed a video in a web page.

Example:

```
<video width="320" height="240" controls>  
<source src="movie.mp4" type="video/mp4">  
<source src="movie.ogg" type="video/ogg">  
</video>
```

