CS511

Notes for mid-term

by "Muhammadians"

For Your kind information (read well):

Asslam-o-Alaikum!

Virtual university's students ***"Junaid Waris and Farhan khan"*** are here. As there are no official handouts given by the virtual university, the university provides a **reading section** after every module. We've just copied that and pasted text as well as images. After completion of the midterm syllabus, it will be named **"midterm notes of CS511 by VU."**.

Now this is the pdf file of the first **four** weeks only. Because still only four weeks are opened. These notes will **be** updated every week until midterm and shared in groups. Only this file will be updated and shared in groups. Before midterm exams, this will update completely. Try to find out where you got these notes the first time, and if you don't find them, then you can contact us in our **official group** (the group link is given below). And again, final-term notes for CS511 will upload after mid-term exams.

We hope you will find it best because it's difficult to read every module's reading material from LMS, so we tried to save your time and headache as well. We also tried to mark short/long questions and quizzes abbreviated as **(SQ, LQ, MCQ, respectively)** as we could, but not totally**.** And sometimes if the quizzes and short questions are in the topic's explanation, then **boldness** shows them. We compiled this for the welfare of virtual university students. All credits go to only Virtual University, who provide us the material.

Don't print out this pdf file. This file has half blank pages or 3rd of 4th blank page (because of module ending). Only going through, just reading bold material, and remembering only SQ, LQ, and quizzes; these do not ensure the preparation for good marks, although they do ensure passing marks. Well, I wish you the best of luck for your future! Remember us in your prayers. Thank you so much!

Regards:

- © "Junaid Waris"
- © "Farhan Khan"

WhatsApp Group Link:

https://chat.whatsapp.com/GkX5GZnrdnWDDwWk4QB1ec

(We may respond slowly but respond to you effectively.)

Week #01

(Modules 1-11)

Module 01:

This course has been divided in two parts. First part deals with basic topics and second part deals with advance topics.

Outline of this course is as follows:

Topics (Basics)

1. What is web and how does it work?
2. Structuring web contents using HTML

   .. Initial learning of HTML will be included.
3. Setting visual layout using CSS

   .. CSS deals with **coloring and font styling** of web page**. (MCQ)**
4. Making web interactive using JavaScript

   . Java script also being used for full **scale development** these days.
5. Web backend programming using PHP and databases

   .PHP is used for **dynamic web** applications

Topics (Advanced)

6. Engineering web application

   .. study of different architectures and pattern
7. Making websites secure

   .. to avoid the hacking attacks and other vulnerabilities
8. Introduction to web services

   .. to perform some functions and task performed on remote server
9. Front-end and back-end frameworks

   .. nodejs, react, angular, view

10. Hosting websites

11. Search engine optimization (SEO) (MCQ)

.. how to make your website on top position of search engine results.

Textbook

"Fundamentals of Web Development", 2nd Edition by Randy Connolly and Ricardo Hoar, 2018, Pearson

Reference websites

https://www.w3schools.com/

https://developer.mozilla.org/en-US/

https://reactjs.org/

Credits

Reading Material and figures are provided by Pearson Education.

Reference websites

https://www.w3schools.com/

https://developer.mozilla.org/en-US/

https://reactjs.org/

Module 02:

A Complicated Ecosystem

As visualized in following figure, web development can also be understood as an ecosystem, one that builds on existing **technologies (URL, DNS, and Internet)**, and contributes new **protocols and standards (HTTP, HTML, and JavaScript)** that facilitate client-server interactions. As this ecosystem matures, new client and server technologies, frameworks, and platforms continue to be developed in support of the **web (PHP, jQuery, Bootstrap, etc.).**



The model shows web-development as a three-story building. Each floor depicts several aspects of web development with an illustration. Ground floor shows Servers, Configuration, Networks, and Protocols which are the basic building blocks. Middle floor shows different languages which are a part of web development, like CSS, HTML, PHP, JavaScript, databases, APIs, and Tools. Top level shows advanced concepts like Design, Search, Integration, Frameworks, and Security.

A Short History of the Internet

- Telephone Network

  . Telephone networks provide the basic infrastructure.

- Packet Networks
- ARPANET (1969) (**MCQ)**

. This early ARPANET network was funded and controlled by the United States government, and was used exclusively for **academic and scientific purposes**. The early network started small, with just a handful of connected university campuses and research institutions and companies in 1969.

- X.25 (1974) (**MCQ)**

. At the same time, alternative networks were created like X.25 in 1974, which allowed (and encouraged) business use.

- USENET (1979) (**MCQ)**

. USENET, built in 1979, had fewer restrictions still, and as a result grew quickly to 550 connected machines by 1981. It focuses on **unrestricted access**.

- TCP/IP (1983) ⇓ INTERNET (**MCQ + SQ)**

. To promote the growth and unification of the disparate networks, a suite of protocols was invented to unify the networks**. A protocol is the name given to a formal set of publicly available rules that manage data exchange between two points.** Communications protocols allow any two computers to talk to one another, so long as they implement the protocol. By 1981, protocols for the Internet were published and ready for use. New networks built in the United States began to adopt the TCP**/IP (Transmission Control Protocol/Internet Protocol)** communication model.

<u>Internet vs. (World Wide) Web</u> (MCQ + SQ)



"**Internet is the hardware part - it is a collection of computer networks connected through either copper wires, fiber-optic cables or wireless connections** whereas, the **World Wide Web (www)can be termed as the software part – it is a collection of web pages connected through hyperlinks and URLs**. In short, the World Wide Web is one of the services provided by the Internet. Other services over the Internet include e-mail, chat and file transfer services. All of

these services can be provided to consumers for use by businesses or government or by individuals creating their own networks or platforms."

…[https://www.diffen.com/difference/Internet_vs_World_Wide_Web]

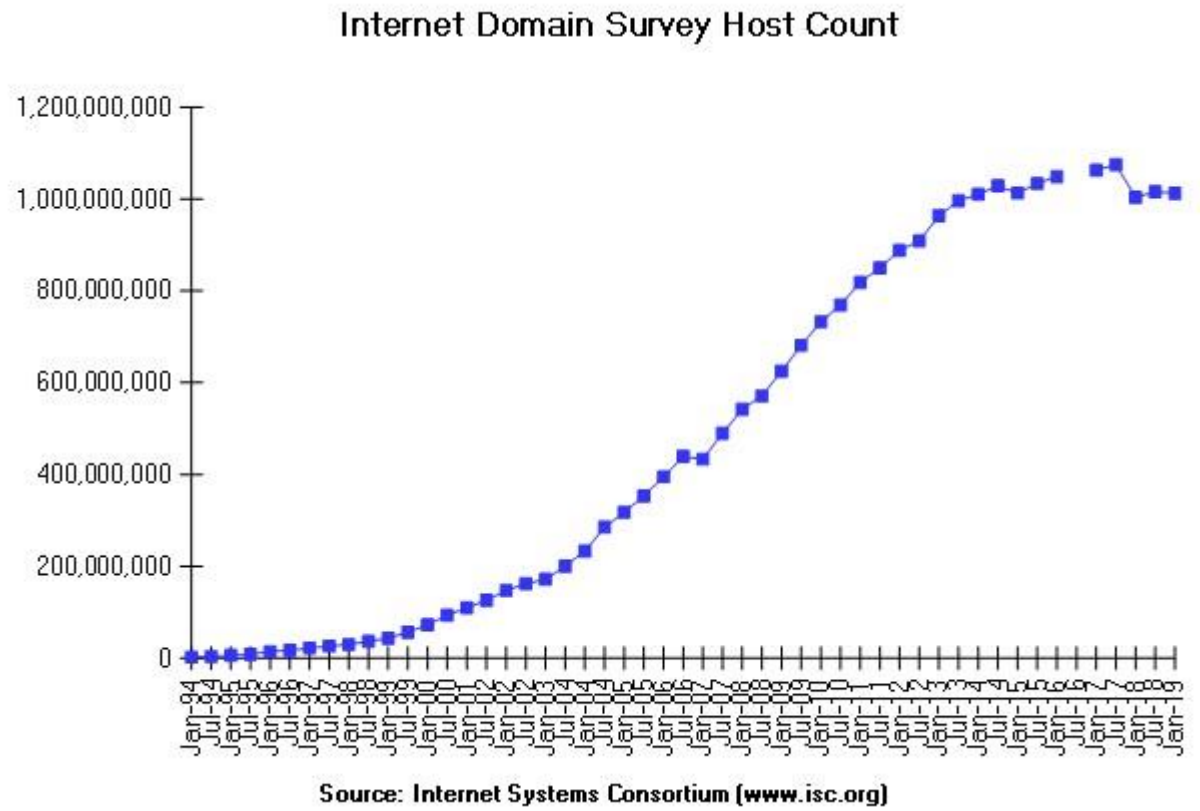
The Birth of the Web (1990) (MCQ)

The British Tim Berners-Lee (now Sir Tim Berners-Lee), who, along with the Belgian Robert Cailliau, published a proposal in 1990 for a hypertext system while both were working at **CERN (European Organization for Nuclear Research)** in Switzerland. Shortly thereafter Berners-Lee developed the main features of the web. This early web incorporated the following essential elements that are still the core features of the web today:

A **Uniform Resource Locator (URL)** to uniquely identify a resource on the WWW. The **Hypertext Transfer Protocol (HTTP)** to describe how requests and responses operate. A software program (later called web server software) that can respond to HTTP requests. **Hypertext Markup Language (HTML) to publish documents**. A program (later called a browser) that can make HTTP requests to URLs and that can display the HTML it receives.

Growth of Internet (MCQ)

Number of Internet Hosts increases tremendously between 2000 to 2015 for a figure of 1 billion hosts.



Web Apps vs. Desktop Apps

- Advantages
- Accessible from any Internet-enabled computer
- Cross platform and browser compatibility
- Easier to roll out program updates
- Fewer security concerns about local storage
- Disadvantages
- Requires internet
- Security concerns about sensitive private data being transmitted over the Internet
- Concerns over the storage, licensing, and use of uploaded data
- Problems with certain websites not having an identical appearance across all browsers
- Restrictions on access to operating system resources
- May need plugins for functioning

Textbook

"Fundamentals of Web Development", 2nd Edition by Randy Connolly and Ricardo Hoar, 2018, Pearson

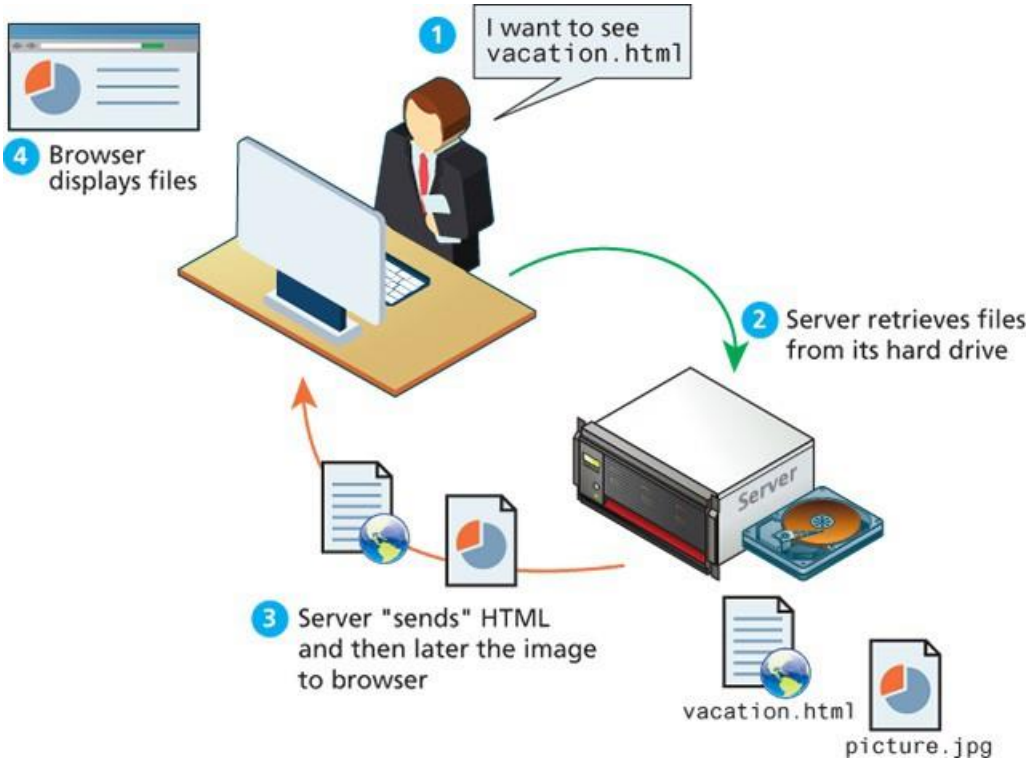
Credits

Reading Material and figures are provided by Pearson Education.

Module 03:

Static website (MCQ + SQ)

Static website consists only of HTML pages that look identical for all users at all times. A publisher would publish web pages and periodically update them. Users could read the pages but could not provide feedback. Following figure illustrates a simplified representation of the interaction between a user and a static website.

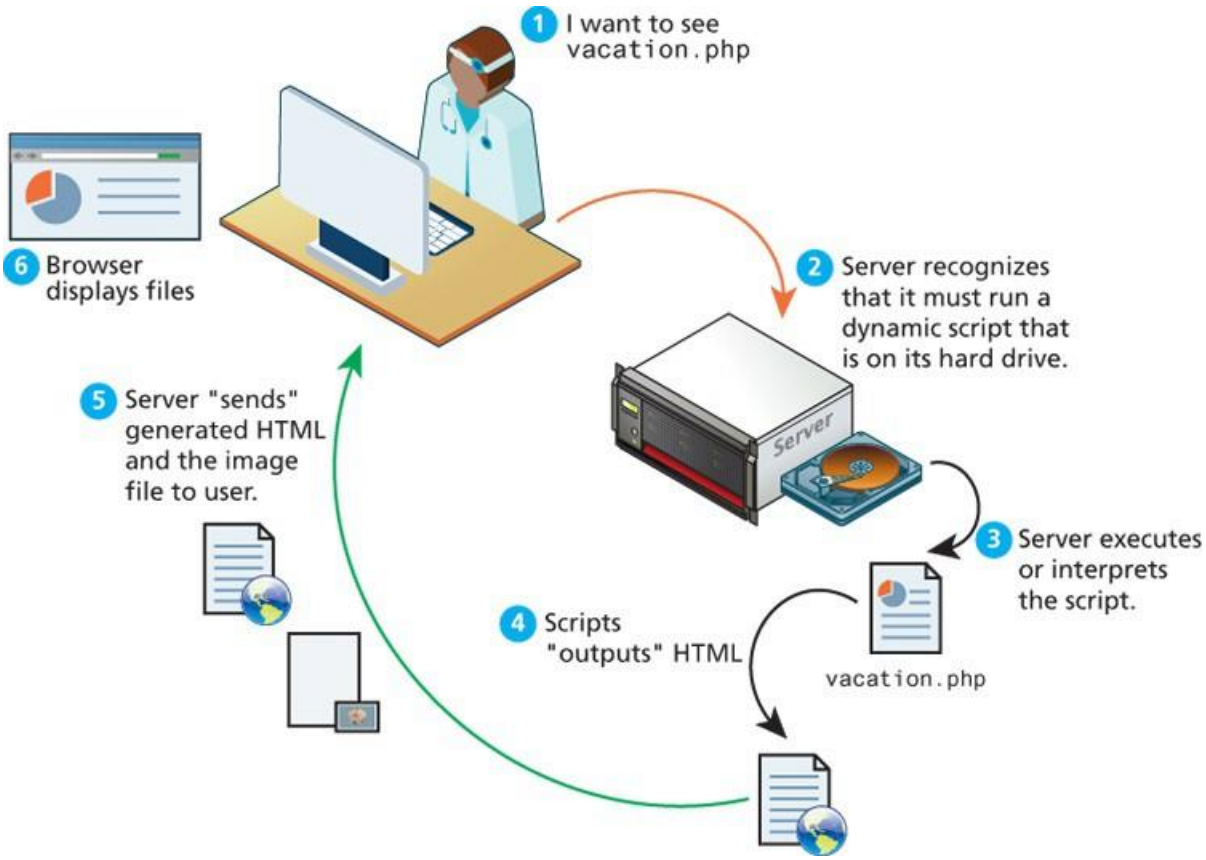


The illustration shows user at a desktop monitor saying, “I want to see vacation.html”. An arrow points from the user to a server machine which has its hard drive open. Server has second step defined as, “Server retrieves files from its hard drive”. Two files labeled “vacation.html” and “picture.jpg” are shown next to the hard drive. This leads to third step, depicting two files being transferred from server to the user's machine. This step is labeled as, “Server 'sends' HTML and then later the image to browser”. This leads to final and fourth step, which shows a webpage consisting a picture and text. Picture and text depicted on webpage is same as depicted on “vacation.html”, and “picture.jpg” files. This step is labeled as “Browser displays files”.

Dynamic Websites (Traditional or server-side)

Within a few years of the invention of the web, sites began to get more complicated as more and more sites began to use programs running on web servers to generate content dynamically. These **server-based programs would read content from databases, interface with existing**

enterprise computer systems, communicate with financial institutions, and then output HTML that would be sent back to the users' browsers. This type of website is called a dynamic server-side website because the page content is being created at run time by a program created by a programmer; this page content can vary from user to user. Following figure illustrates a very simplified representation of the interaction between a user and a dynamic website.



The illustration shows six steps of the interaction as follows:

1. User at a terminal says, “I want to see vacation.php""

2. Server recognizes that it must run a dynamic script that is on its hard drive (an arrow points from the user terminal to the Server, with an open hard drive).

3. Server executes or interprets the script. (An arrow points from the server to a page titled “vacation.php”

4. Scripts “outputs” HTML (an arrow points from vacation.php to a html script.

5. Server “sends” generated HTML and the image file to user. (an arrow points from html file to user terminal).

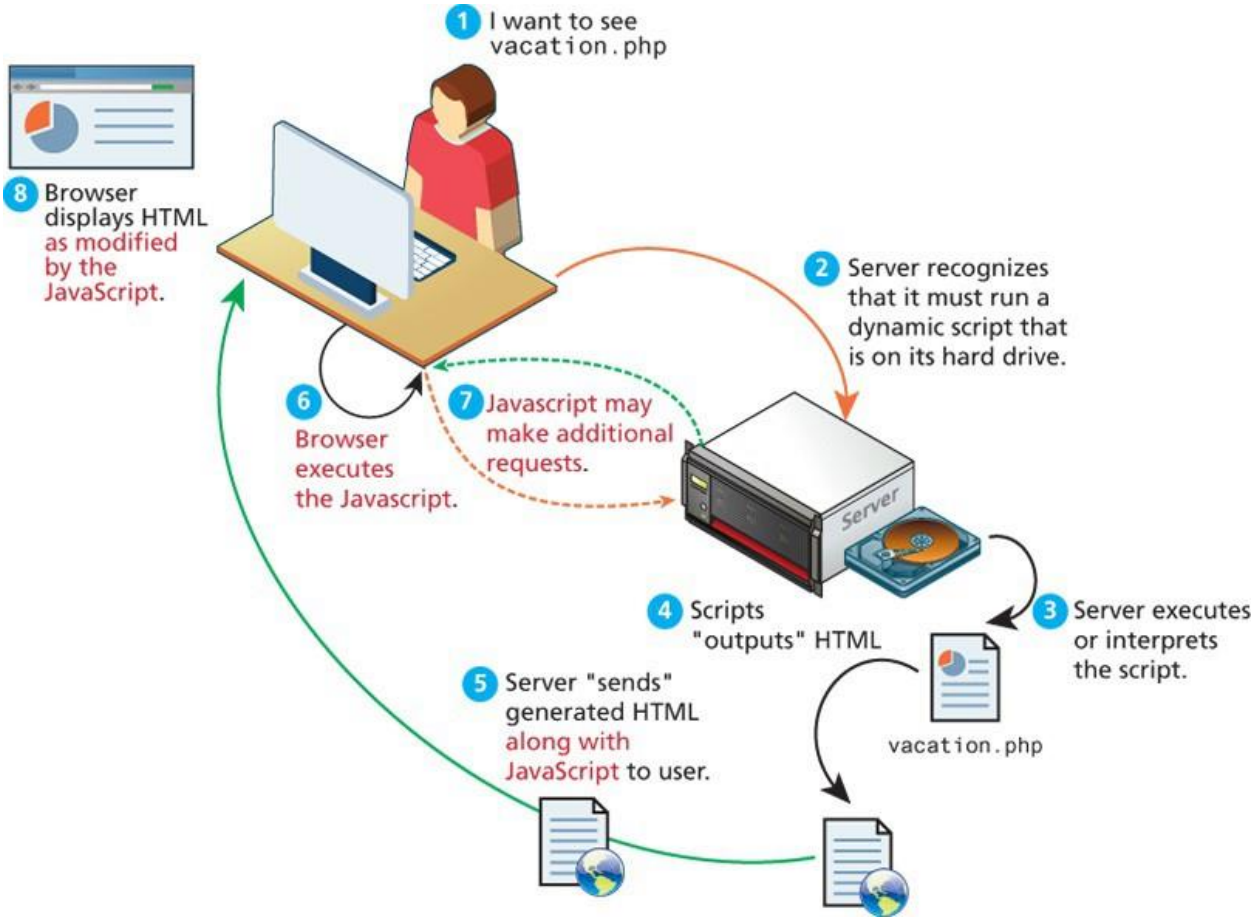
6. Browser displays files (An assembled webpage is displayed on the user's terminal).

Dynamic Websites (Modern)

Dynamic Websites of modern era (Web 2.0) refer to an interactive experience where users could contribute and consume web content, thus creating a more user-driven web experience.

For software developers, (Web 2.0) require more programming logic, which previously existed only on the server, began to migrate to the browser (see following figure).

This required learning **JavaScript**, a rather tricky programming language that runs in the browser, as well as mastering the rather difficult programming techniques **involved in asynchronous communication.**



The illustration shows six steps of interaction between a user and a dynamic website depicted as:

1. User at a terminal says, “I want to see vacation.php”

2. Server recognizes that it must run a dynamic script that is on its hard drive (an arrow points from the user terminal to the Server, with an open hard drive).

3. Server executes or interprets the script. (an arrow points from the server to a page titled “vacation.php”).

4. Scripts “outputs” HTML (an arrow point from vacation.php to an html script).

5. Server “sends” generated HTML and the image file to user. (an arrow points from html file to user terminal).

6. Browser executes the JavaScript. (An arrow points back to the user's terminal).

7. JavaScript may make additional requests (back and forth arrows between the user terminal and the server).

8. Browser displays HTML as modified by the JavaScript (An assembled webpage is displayed on the user's terminal).

Textbook

"Fundamentals of Web Development", 2nd Edition by Randy Connolly and Ricardo Hoar, 2018, Pearson

Credits

Reading Material and figures are provided by Pearson Education.

Module 04:

Web 1.0 (MCQ + SQ)

In the earliest days of the web, a **webmaster** (the term popular in the 1990s for the person who was **responsible for creating and supporting a website**) would publish web pages and periodically update them.

Users could read the pages but could not provide feedback.

- Contents generated by publishers
- Company generated companies

Web 1.0 includes both type:

- **Static websites**
- **Dynamic websites**

Web 2.0 (MCQ +SQ)

Web 2.0 is mainly characterized by:

- **User Generated Contents (UGC)**
- **Dynamic websites**

In the mid-2000s, a new buzzword entered the computer lexicon: Web 2.0.

This term had two meanings, one for users and one for developers. **For the users, Web 2.0** referred to an interactive experience where users could contribute and consume web content, thus creating a more user-driven web experience. Some of the most popular websites today fall into this category: Facebook, YouTube, and Wikipedia. This shift to allow **feedback** from the user, such as comments on a story, threads in a message board, or a profile on a social networking site has revolutionized what it means to use a web application.

For software developers, Web 2.0 also referred to a **change** in the paradigm of how dynamic websites are created. Programming logic, which previously existed only on the server, began to migrate to the browser. This required learning JavaScript, a rather tricky programming language that runs in the browser, as well as mastering the rather difficult programming techniques involved in asynchronous communication.

Web 3.0

There is no standard consensus over Web 3.0 but some people have, however, argued that **Web 3.0** will be something **called the semantic web**. It is mainly characterized by:

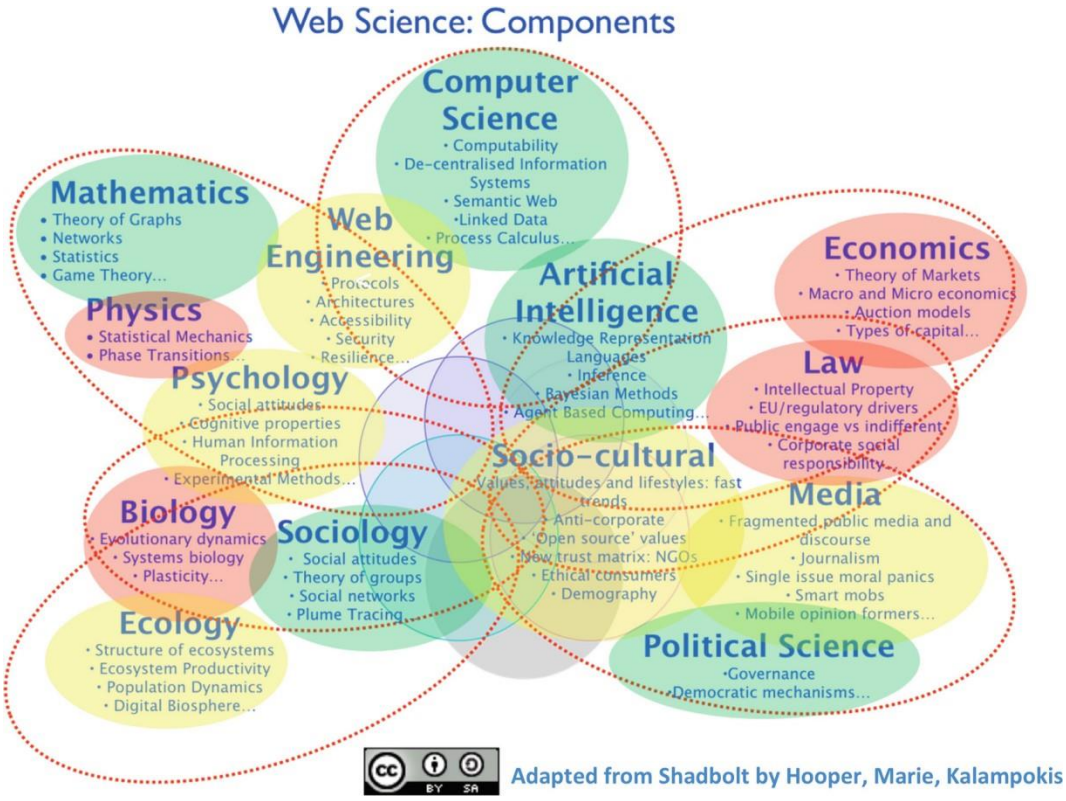
- **Machine understandable contents**
- **Semantic web**

Semantic is a word from linguistics that means, quite literally, **"meaning."** The semantic web thus adds context and meaning to web pages in the form of special markup. These semantic elements would allow search engines and other data-mining agents to make sense of the content. The goal of the semantic web is to make it easier to figure out those meanings, thereby dramatically improving the nature of search on the web.

Web Science

- Interdisciplinary study of sociotechnical integration of the world wide web.

Web Science, as it is known, **studies the sociotechnical** systems that apply the web in areas as diverse as finance, politics, activism, romance, and hate speech. If you look at each interaction on the web as more than just a technical exchange using protocols and file transmission, you can see there is often an underlying social need motivating each exchange. The technical system facilitates a social interaction and social interactions span nearly the entire human experience, so there is now an entire area of study looking at the web as a sociotechnical system. This is just another example of how the web can facilitate entirely new areas of study.



Textbook

"Fundamentals of Web Development", 2nd Edition by Randy Connolly and Ricardo Hoar, 2018, Pearson

Credits

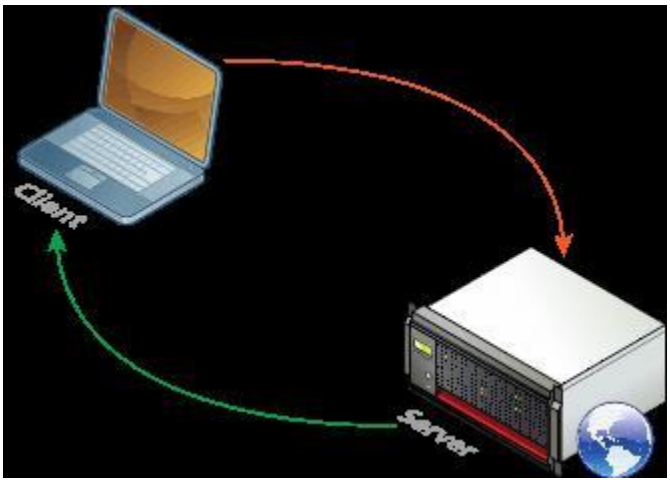
Reading Material and figures are provided by Pearson Education.

Module 05:

Client server model (MCQ + SQ)

The web is sometimes referred to as a client server model of communications. **In the client-server model, there are two types of actors: clients and servers.** The server is a computer agent that is normally active 24/7, listening for requests from clients. A client is a computer agent that makes requests and receives responses from the server.

The request-response Loop



The diagram shows a laptop labelled as “client”. An arrow, labelled as “Request”, is pointing from the laptop to a server machine, depicted with a blue globe next to it. This server machine is labelled as “server”. Another arrow from the server machine to the laptop is labelled as “Response”.

Server Types (SQ: write the names of server types. LQ: also write working.)

Most real-world websites are typically not served from a single server machine, but by many server machines. It is common to split the functionality of a website between several different types of servers, as shown in following figure. These include the following:

Web servers: (MCQ + SQ)

A web server is a computer servicing HTTP requests. This typically refers to a computer running web server software, such as Apache or Microsoft IIS (Internet Information Services).

Application servers: (SQ)

An application server is a computer that hosts and executes web applications, which may be created in PHP, ASP.NET, Ruby on Rails, or some other web development technology.

Database servers: (SQ)

A database server is a computer that is devoted to running a Database Management System (DBMS), such as MySQL, Oracle, or MongoDB, that is being used by web applications.

Mail servers: (SQ + MCQ)

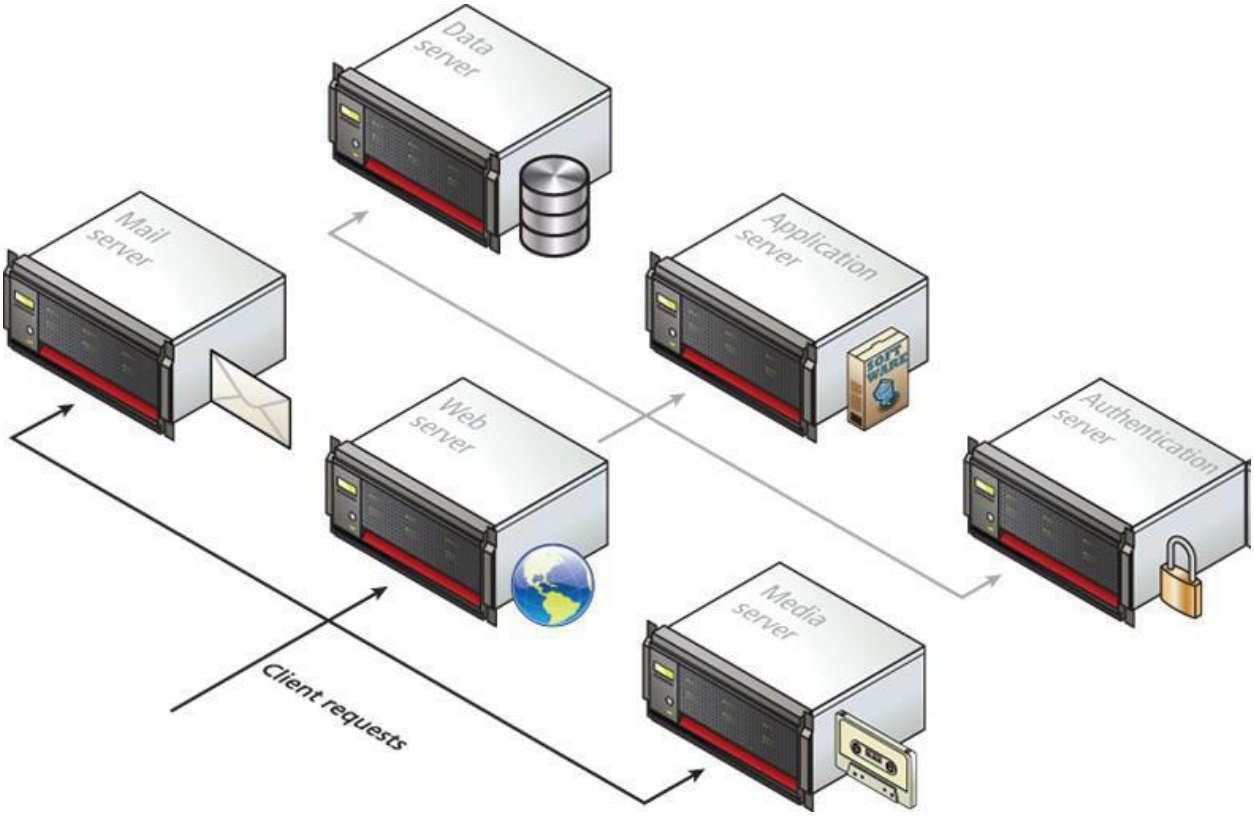
A mail server is a computer creating and satisfying mail requests, typically using the **Simple Mail Transfer Protocol (SMTP).**

Media servers: (SQ+ MCQ)

A media server (also called a streaming server) is a special type of server dedicated to servicing requests for images and videos. It may run special software that allows video content to be streamed to clients.

Authentication servers: (SQ + MCQ))

An authentication server handles the most common security needs of web applications. This may involve interacting with local networking resources, such as **LDAP (Lightweight Directory Access Protocol)** or Active Directory.

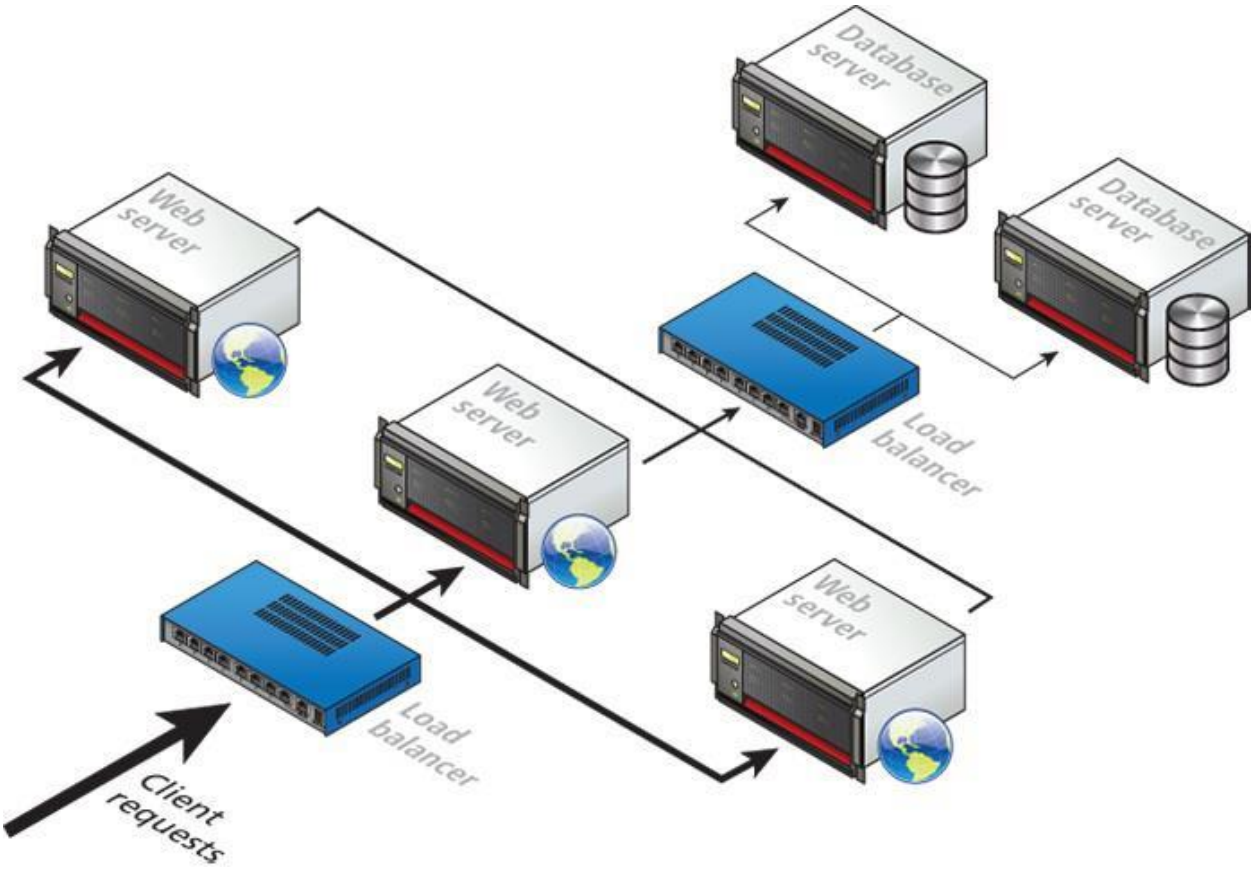


In smaller sites, these specialty servers are often the same machine as the web server.

Server Farms: (MCQ)

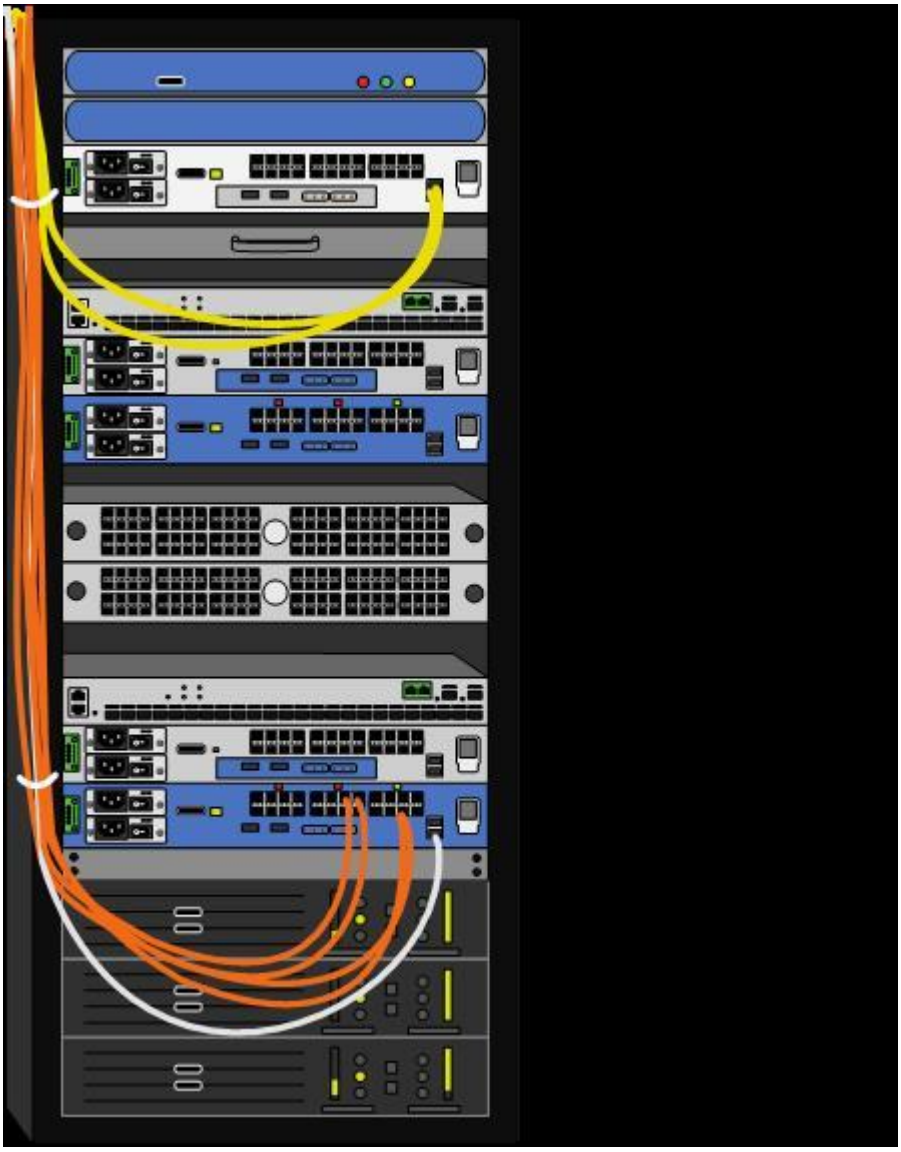
A single web server that is also acting as an application or database server will be hard-pressed to handle more than a few hundred requests a second. A busy site can receive thousands or even tens of thousands of requests a second; globally popular sites such as Facebook receive millions of requests a second.

The usual strategy for busier sites is to use a server farm. The goal behind server farms is to distribute incoming requests between clusters of machines so that any given web or data server is not excessively overloaded, as shown in following figure. **Special devices called load balancers distribute incoming requests to available machines.**



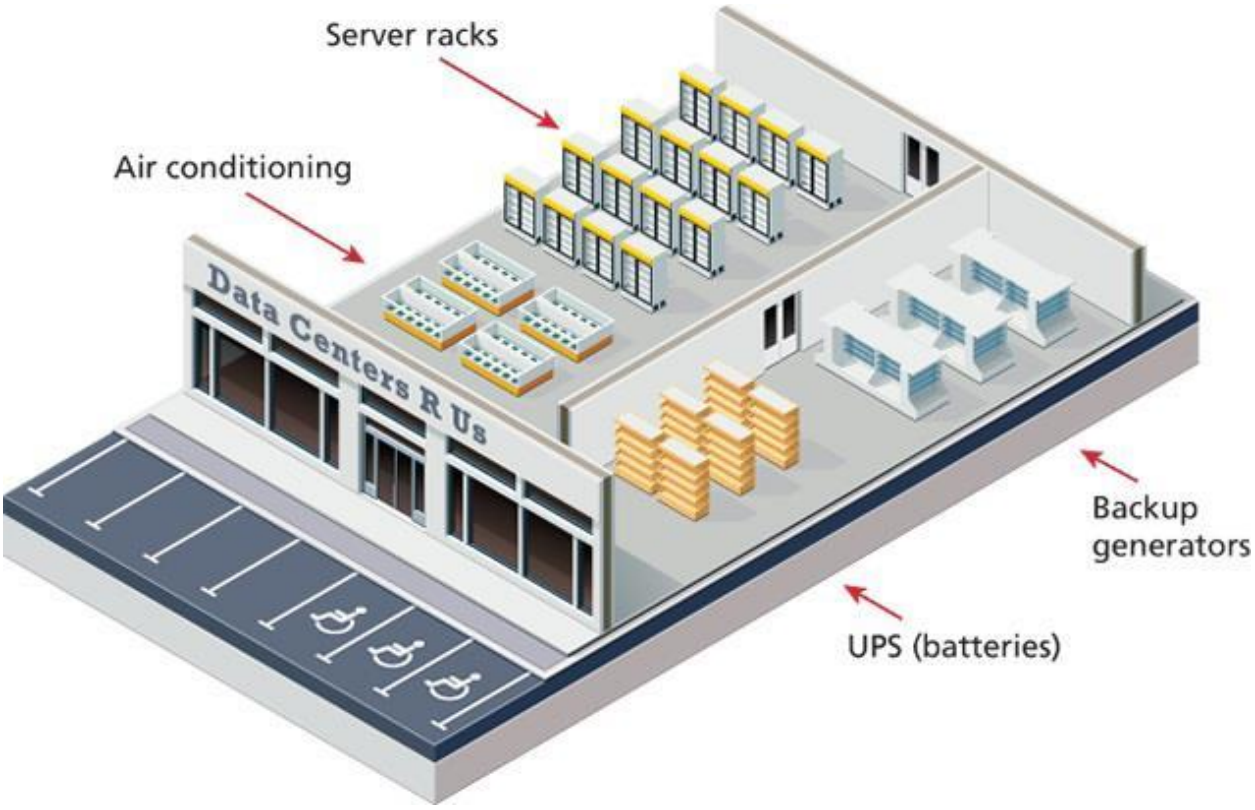
Server Racks:

In a server farm, the computers do not look like the ones in your house. Instead, these computers are more like the plates stacked in your kitchen cabinets. That is, a farm will have its servers and hard drives stacked on top of each other in server racks. **A typical server farm will consist of many server racks, each containing many servers**, as shown in following figure.



Data Centers (SQ + MCQ) imp.

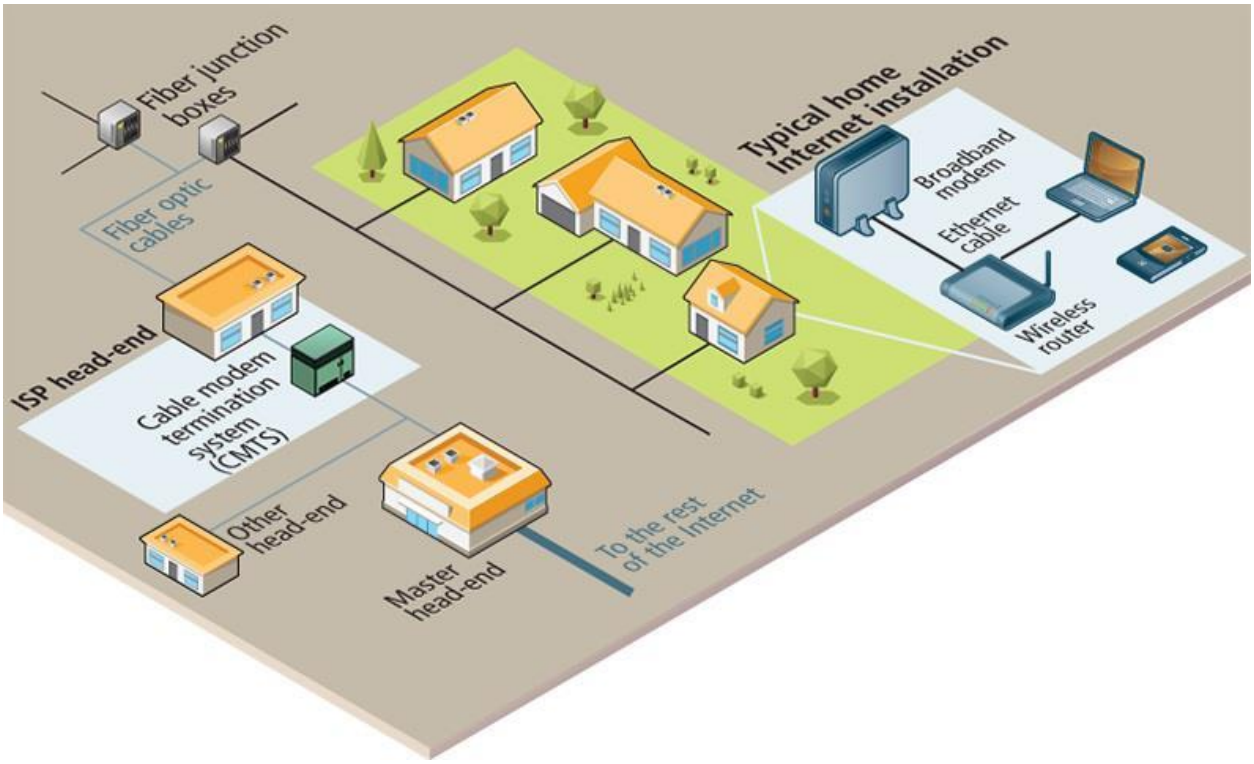
Server farms are typically housed in special facilities called data centers. A data center will contain more than just computers and hard drives; sophisticated air conditioning systems, redundancy power systems using batteries and generators, specialized fire suppression systems, and security personnel are all part of a typical data center, as shown in following figure.


Server racks
Air conditioning
Data Centers R Us
Backup generators
UPS (batteries)

Module 06:

Internet from Home to ISP:(MCQ + SQ)

While there are many configuration possibilities, following figure does provide an approximate simplification of a typical home to local **Internet Server Provider (ISP)** setup.



The broadband modem, also called a cable modem or DSL (digital subscriber line) modem, is a bridge between the network hardware outside the house (typically controlled by a phone or cable company) and the network hardware inside the house. The wireless router is perhaps the most visible manifestation of the Internet in one's home.

The various neighborhood broadband cables are aggregated and connected to fiber optic cable via fiber connection boxes.

These fiber optic cables eventually make their way to an ISP's head-end, which is a facility that may contain a **cable modem termination system (CMTS)** or a **digital subscriber line access multiplexer (DSLAM)** in a DSL based system. This is a special type of very large router that connects and aggregates subscriber connections to the larger Internet. These different head ends may connect directly to the wider Internet, or instead be connected to a master head-end, which provides the connection to the rest of the Internet.

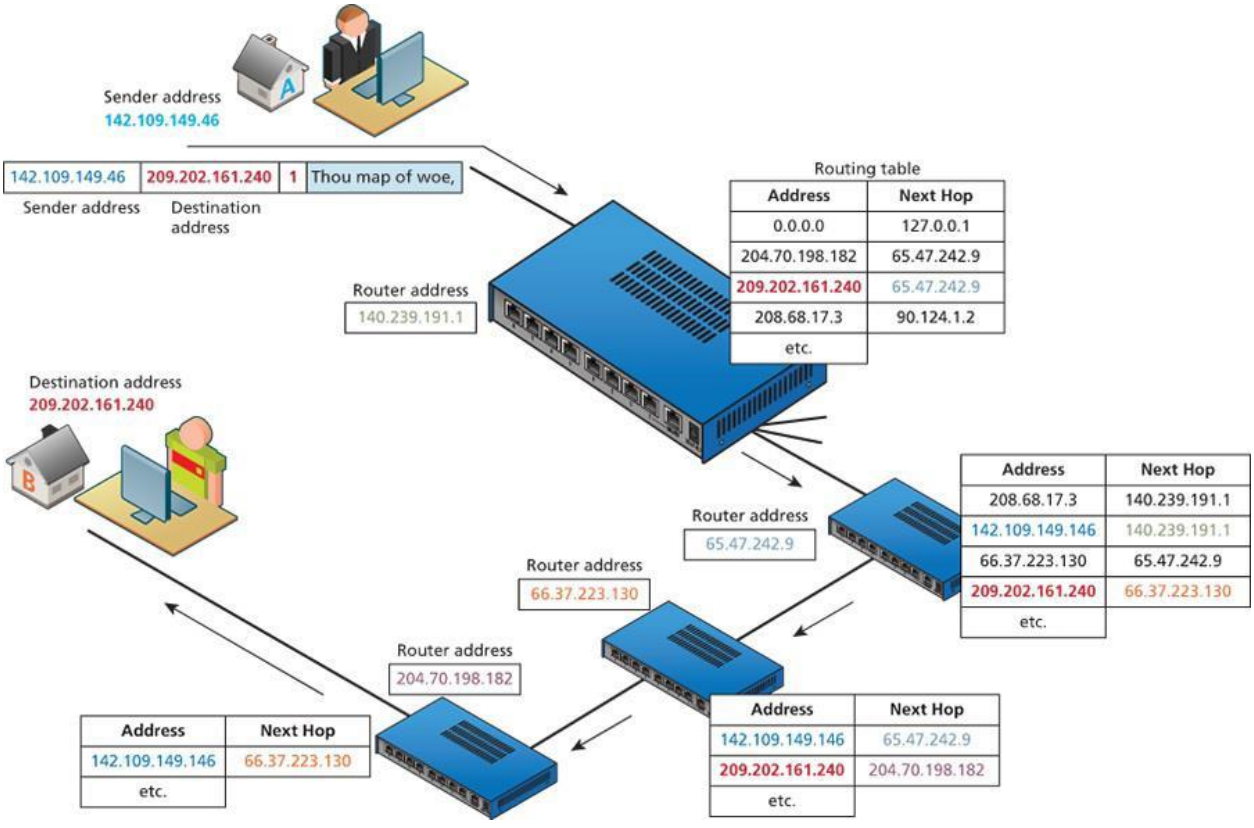
Routing Tables (SQ+ MCQ +LQ: Describe working of router.)

The flow of internet traffic or information is carried out mainly by routers**. A router is a hardware device that forwards data packets from one network to another network.** When the router receives a data packet, it examines the packet's destination address and then forwards it to another destination. A router uses a routing table to help determine where a packet should be sent. It is a table of connections between target addresses and the destination (typically another router) to which the router can deliver the packet. In following figure, the different routing tables use next-hop routing, in which the router only knows the address of the next step of the path to the destination; it leaves it to that next step to continue the routing process. The packet thus makes a variety of successive hops until it reaches its destination.

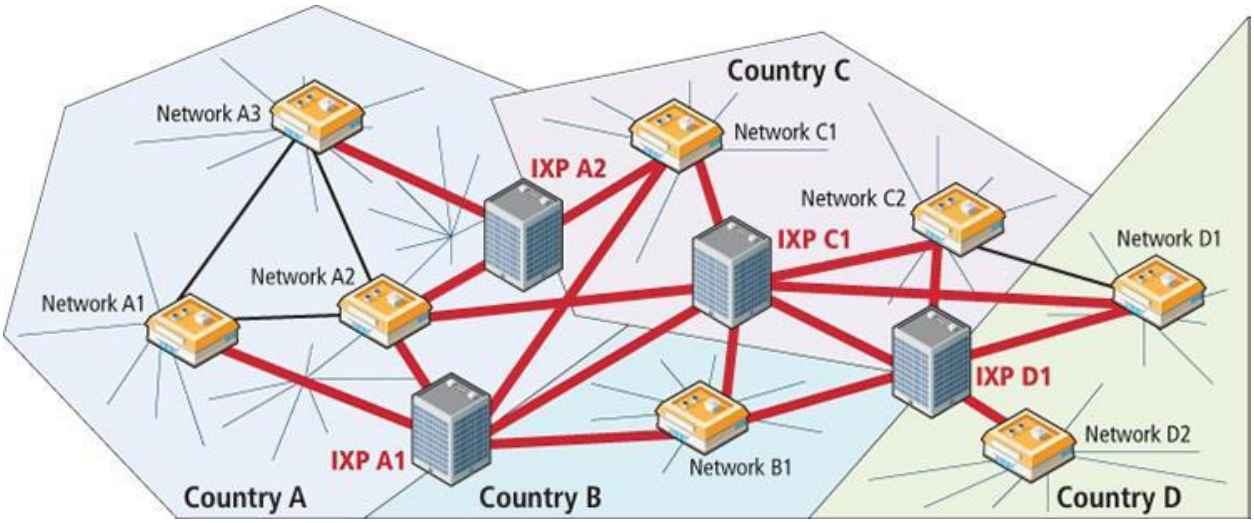
From ISP to the World:

Internet exchange point (IX or IXP)

Dedicated infrastructure to make the internet more connected. Internet exchange point (IX or IXP) provides a way for networks

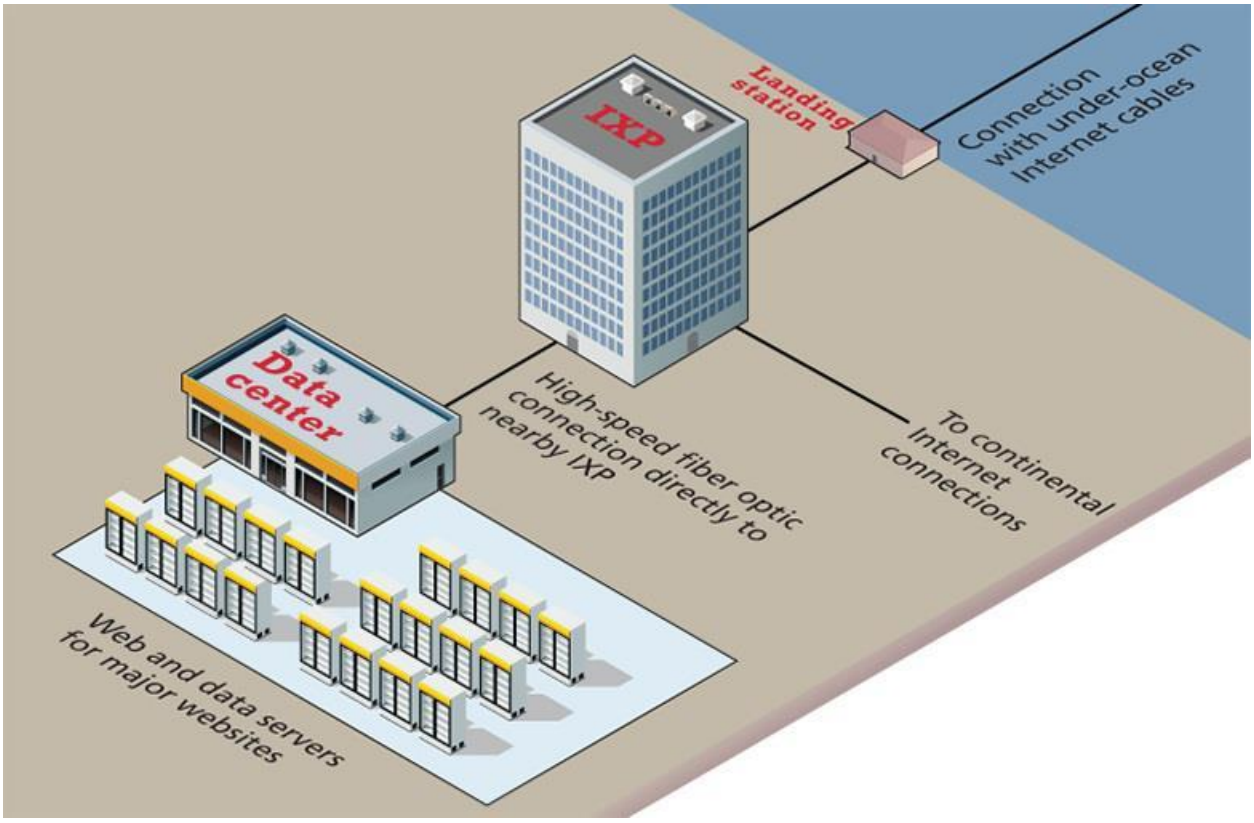


within a country to interconnect and between different countries. More and more networks began to interconnect with each other using an Internet exchange point (IX or IXP). Multiple paths between IXPs provide a powerful way to handle outages and keep packets flowing. Another key strength of IXPs is that they provide an easy way for networks to connect to many other networks at a single location.



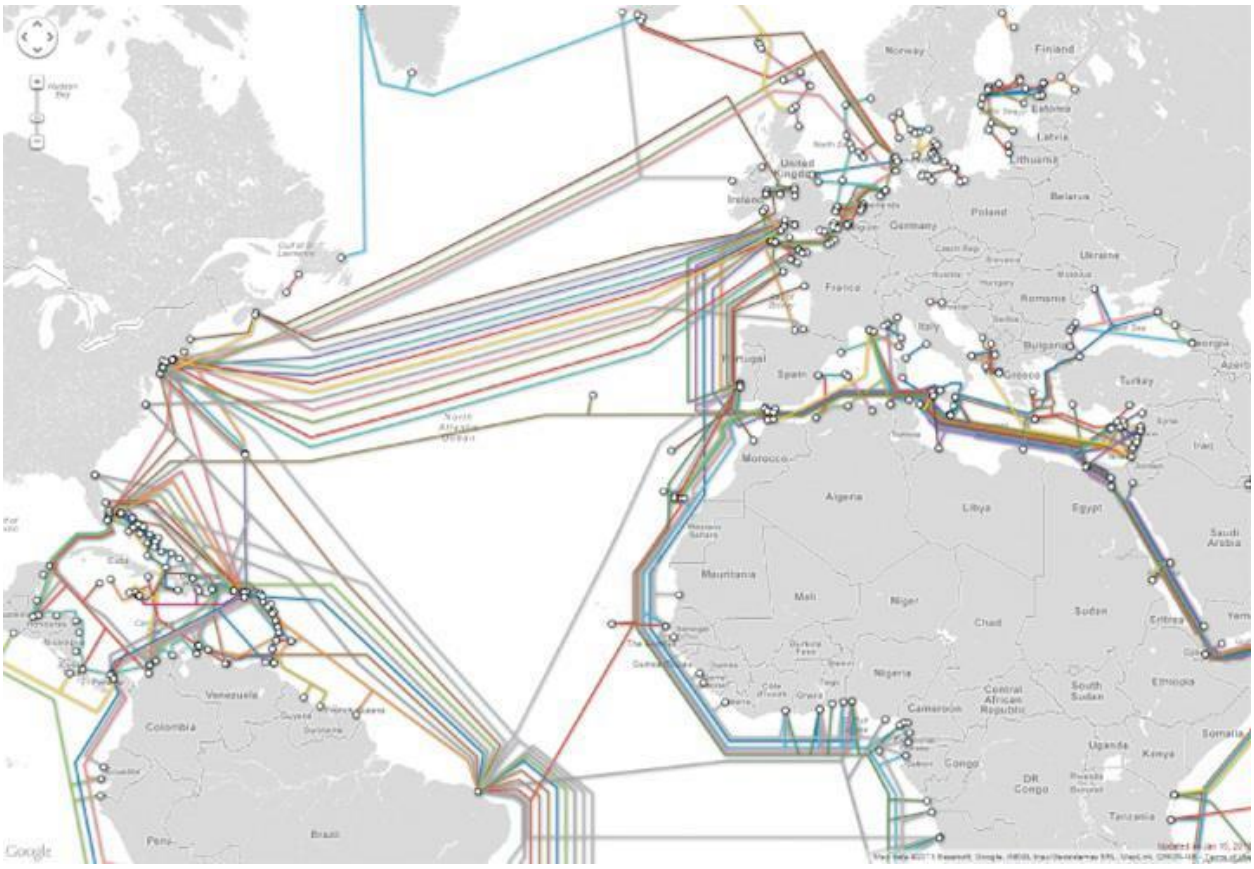
IXPs and data centers

In recent years, major web companies have joined the network companies in making use of IXPs. As shown in following figure, this sometimes involves mirroring (duplicating) a site's infrastructure (i.e., web and data servers) in a data center located near the IXP. The website will have incremental speed enhancements (by reducing the travel distance for these sites) across all the networks it is peered with at the IXP, while **the network will have improved performance for its customers when they visit the most popular websites. (MCQ)**



Under-ocean connectivity (MCQ)

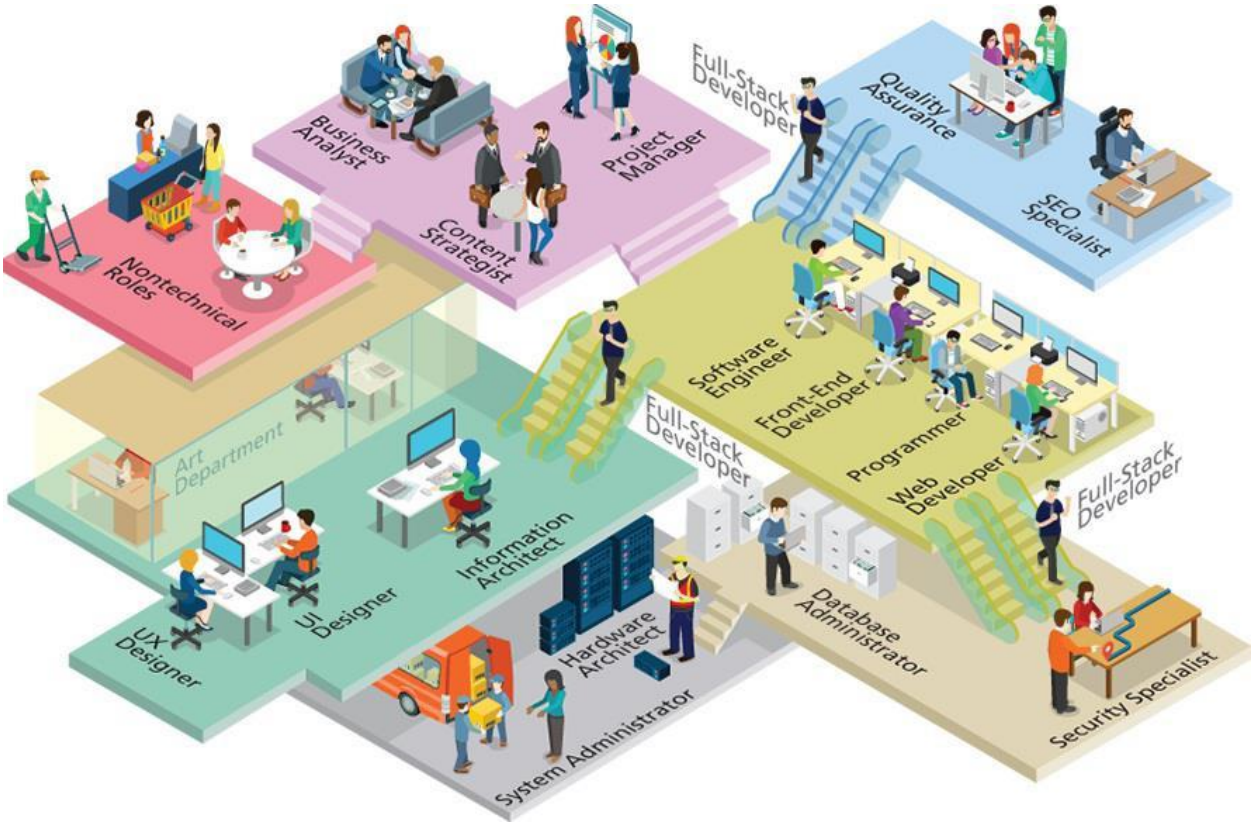
Eventually, international Internet communication will need to travel underwater. The amount of **undersea fiber optic cable** is quite staggering and is growing yearly. As can be seen in following figure, over 250 undersea fiber optic cable systems operated by a variety of different companies span the globe.



Module 07:

Web Engineering Jobs

Following figure illustrates the complex ecosystem that is contemporary web development. Seeing that diagram, you should learn that there are many different jobs that one can do within the web development world.



<u>Roles and Skills</u> (LQ: Roles of jobs, SQ: describe role of someone specific)

Following list of job titles provides an overview of the roles typically available in a web development company as part of a team.

<u>Hardware Architect/Network Architect/Systems Engineer</u> (SQ)

The people who design the specifications for the servers in a data center, and design and manage the layout of the physical and logical network are essential somewhere along the way, whether at your company or your host's. Typically, these roles require networking and operating systems knowledge.

<u>System Administrator</u>(SQ)

Once the system is built and wired to the network, system administrators are the next people required to get things up and running. Often, they choose and install the network operating

system, then manage the shared operating system environments for other users. This position is often combined with the hardware architect in smaller firms.

Database Administrator/Data Architect (SQ)

The **database administrator (DBA)** is a role found in larger companies. In these companies, there are many databases, often from many divisions, all of which need to be managed, secured, and backed up**. Database administrators will perform maintenance** on the databases as well as manage access for user and software accounts. They sometimes write triggers and advanced queries for users upon request as well as manage database indexes.

A **data architect** has some overlap with database administrator, but the role is more focused on the **design and integration of data**. In recent years, managing and making use of large sets of often unrelated data has become increasingly important for web companies. In smaller companies, these different data roles are often combined with the system administrator and/or developer ones.

Security Specialist/Consultant/Expert (SQ)

A good system administrator and network architect will certainly have insights into security as they perform their duties. However, because security is so vital to web development in general, and because the knowledge necessary to do security work is complicated and ever changing, it is not uncommon for companies to outsource their security needs to security specialists. These specialists will test for vulnerabilities, implement security best practices, and make updates and changes to programming code or hardware infrastructure to protect a site against well-known or newly emerging (called zero-hour) threats.

Developer/Programmer (SQ)

Programmers can be assigned a wide range of tasks aside from simple coding. Writing good documentation, using version control software, engaging in code reviews, running test cases, and more might be typical tasks, depending on company practices. Programmer positions often begin at the entry level, with higher-level design decisions left to software engineers and senior developers. In terms of the web development world, the terms programmers or developer are quite broad; typically, however, this term is used to indicate a job focused more on server-side development using languages like PHP.

Front-End Developer/UX Developer (SQ)

Increasingly complex front-end development requires software developers with an aptitude for **graphical user interface(GUI)** design (nowadays more typically referred to as user-experience or UX design) and an understanding of **human–computer interaction (HCI)** principals. This typically requires in-depth JavaScript expertise along with good CSS skills. Another increasingly commonly used **synonym for front-end developer is UX developer**. The main difference between a UX developer and a UX/UI designer (described below), is that **the UX developer is involved mainly in the implementation of the user experience and less in the actual design of it.**

Software Engineer (SQ)

"A software engineer is a programmer who is adept at the language of analysis and design, and uses established best practices in the development of software."

Sometimes the role of a programmer and software engineer are used interchangeably, but a software engineer has more knowledge of the software development life cycle and can effectively gather requirements and speak with clients about technical and business matters.

UX Designer/UI Designer/Information Architect (SQ)

These are names used somewhat interchangeably for jobs that focus on the structure, design and usability of a website. Once referred to as the user interface, the term UX has become the preferred term because improving how a website is used is just as important (or even more important) nowadays as improving how a website appears. While coding skills can be helpful, this type of work more often involves the development of prototypes, making mockup designs, and analyzing user experience data. In larger web development firms, this type of work also commonly involves working in conjunction with creatives in the art department.

Tester/Quality Assurance (SQ)

"Testers are the people who try to identify flaws in software before it gets released." This type of work is often called **quality assurance (QA).** Although some test roles are for nonexperts, many testers know how to program and might write automated tests as well as develop testing plans from requirements. Although these duties are often integrated with developers, they can form a job all their own.

SEO Specialist (SQ)

Search engine optimization (SEO) refers to the process of improving the discoverability of web content by search engines. Chapter 23 covers both the above board (as well as the under-handed) techniques used to improve SEO results. An SEO specialist needs to be familiar with these techniques as well as analytics, testing approaches, social networking APIs, and even content creation strategies.

Content Strategists/Marketing Technologist (SQ)

Regardless of technological features, websites ultimately succeed due to the quality of their content. A **content strategist** (sometimes also called a **marketing technologist**) is someone who uses his or her experience with existing and emerging web technologies in conjunction with knowledge about the audience to craft engaging web content. This type of work might also be done by an SEO specialist or an information architect. Writing and marketing skills as well as knowledge of content management systems, email services, and social networking interfaces are important for this job.

Project Manager/Product Manager (SQ)

Websites are complicated projects often involving the work of many different people with different skill sets and personalities. Getting all these people to work together in a timely and effective manner typically requires the committed effort and knowledge of project managers (also called product managers). Knowledge of planning and estimation methodologies is helpful, as are more general people management skills.

Business Analyst (SQ)

Although a software engineer in an analysis role might speak to clients and get requirements, that role is often given a different name and assigned to someone with especially good communication skills. A business analyst is the interface between the various divisions of the company and the website (and IT in general). These people can easily speak to the HR, marketing, and legal divisions, and then translate those requirements into tasks that software engineers can take on.

Nontechnical Roles (SQ)

Aside from all the technical roles above, there are additional important roles that require expertise outside of technology. These roles include traditional ones found in almost every company: accountants, writers, designers, editors, lawyers, salespeople, and managers. There are also a wide variety of new roles that are unique to the web space,10 such as analytics manager, motion designer, social media analyst, cloud architect, and the intriguingly named growth hacker. Getting people from different backgrounds with different expertise to work together is how companies balance the business, technology, and art of website development.

Full-stack developer (SQ)

Rather than specializing in server-side development, or client-side user experience construction, or database administration, a full-stack developer ideally has competency and experience in all of these domains. Indeed, many companies even expect full-stack developers to be knowledgeable about various system administration tasks, such as setting up a web server and handling security issues.

Web Development Companies (MCQ+ SQ: names of companies LQ: describe working)

A major factor to consider when thinking about a career in web development is what kind of company you want to work for. Sure, everyone needs a website, but there are multiple kinds of companies that work together to make that a possibility (illustrated in following figure).



Hosting Companies (SQ)

There are companies that will manage servers on your behalf. These hosting companies or data centers offer many employment opportunities, especially related to hardware, networking, and system administration roles.

Design Companies (SQ)

Design companies are at the opposite end of the spectrum, with few technical positions available. These firms will provide professional artistic and design services that might go beyond the web and include logos and branding in general. Some companies produce mockups in Photoshop, for example, which a web developer (at another company) can then turn into a website.

Website Solution Companies (SQ)

Website solution companies focus on the programming and deployment of websites for their clients. There are technical positions to help manage the existing sites (working in conjunction with hosting companies) as well as development jobs to build the latest custom site.

Vertically Integrated Companies (MCQ+SQ)

Vertically integrated companies are increasingly becoming the one-stop shop for web development. **They are called vertically integrated because these companies combine hosting, design, and application solutions into one company**. This allows these companies to achieve economies of scale and appeal to nontechnical clients who can go there for all their web-related needs, large or small.

Start-Up Companies (SQ)

Start-up ventures in web development have been some of the biggest success stories in the business world. Start-ups are often attractive places for new graduates to work, with less competition from experienced candidates and potentially lots of jobs available from developers to designers and system administrators. The smaller start-ups companies often require full-stack developers, who can take on any role from system administrator through to lead developer.

Internal Web Development (SQ)

Although many companies outsource their web presence, others assign the work to an internal division, normally under the umbrella of IT or marketing. Although many of these roles are simple caretaker positions, others can be quite engaging, requiring real programming expertise. Many companies have lots of internal data they would not share with outsiders and thus prefer inhouse expertise for the development of web interfaces and systems to manage and display that confidential data. Often these websites exist only with an organization's Intranet rather than as public websites on the Internet.

Textbook

"Fundamentals of Web Development", 2nd Edition by Randy Connolly and Ricardo Hoar, 2018, Pearson

Credits

Reading Material and figures are provided by Pearson Education.

Module 08:

<u>Internet Protocols</u> (MCQ + SQ)

A protocol is a set of rules that partners use when they communicate.

<u>TCP/IP</u> (MCQ + SQ)

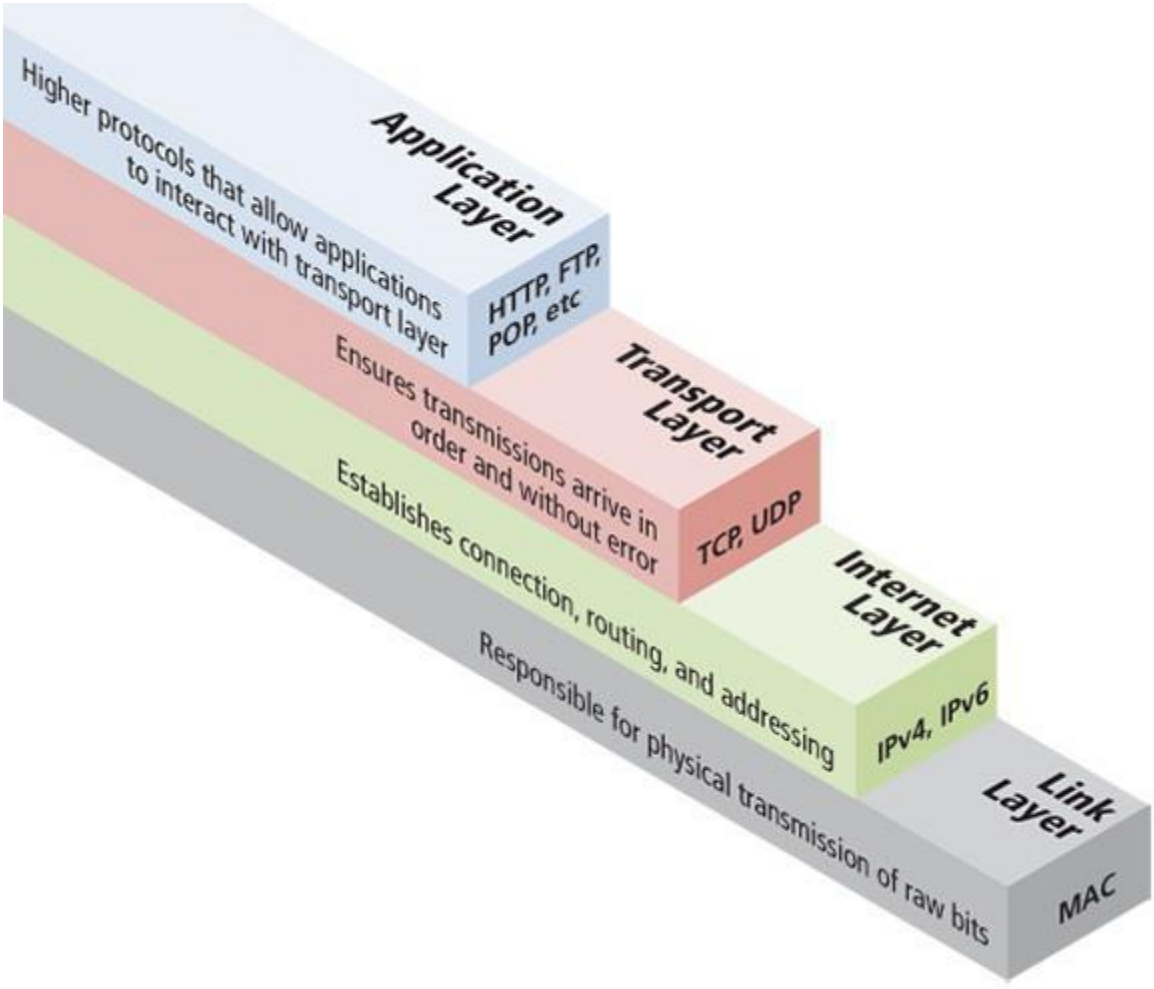
These protocols have been implemented in every operating system, and make fast web development possible.

Networking is its own entire discipline.

Web developer needs general awareness of what the suite of Internet protocols does.

<u>Layered Architecture</u> (MCQ + SQ: write names)

The TCP/IP Internet protocols were originally abstracted as a **four-layer stack**.



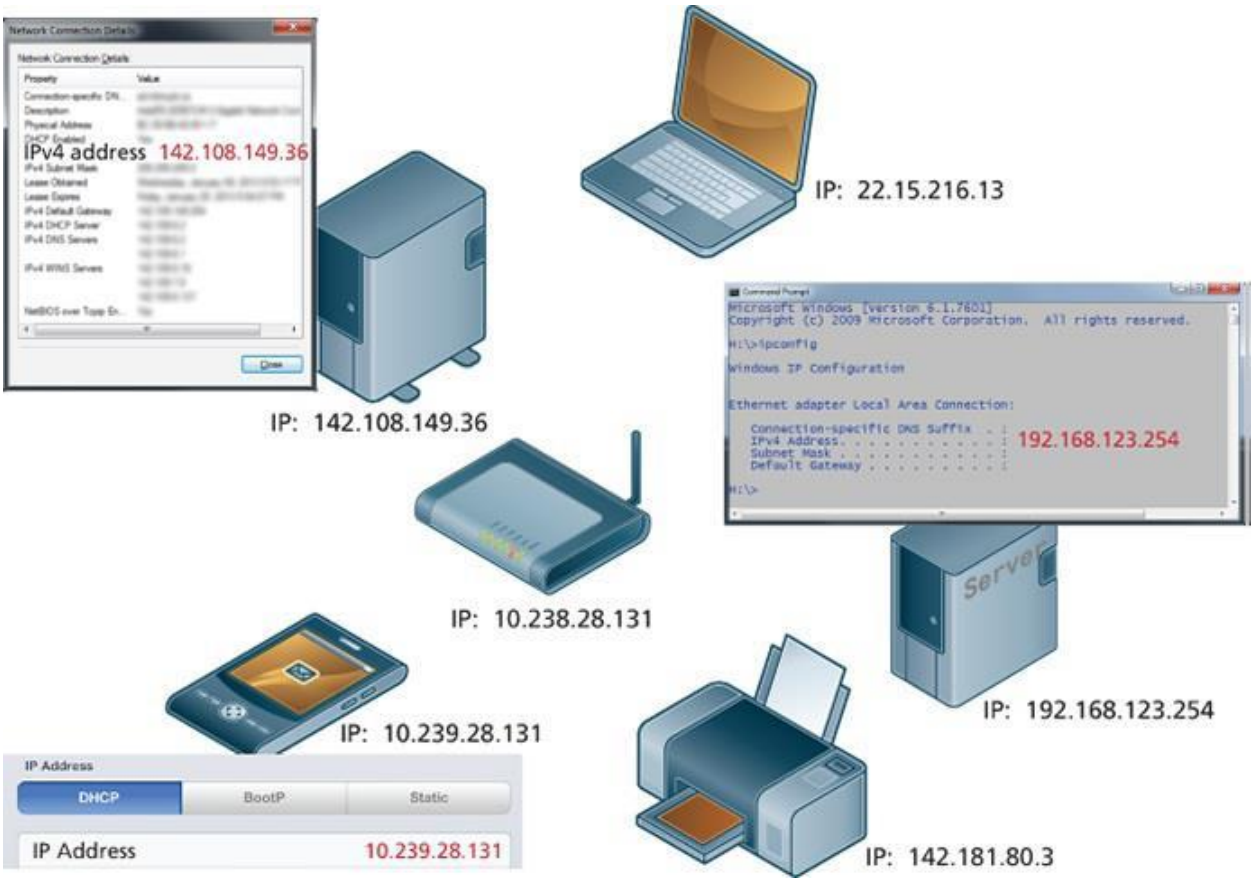
Link Layer (MCQ + SQ)

The link layer is the **lowest layer,** responsible for both the physical transmission of data across media (both wired and wireless in the form of raw bits, light waves etc.) and establishing logical links. **It handles issues like packet creation, transmission, reception, error detection, collisions, line sharing, and more**. **MAC (media access control)** addresses are unique **48- or 64-bit** identifiers assigned to network hardware and which are used at the link layer.

Internet Layer (MCQ + SQ)

The Internet layer (sometimes also called the **IP Layer**) routes packets between communication partners across networks. **The Internet layer provides "best effort" communication**. It sends out a message to its destination, but expects no reply, and provides no guarantee the message will arrive intact, or at all.

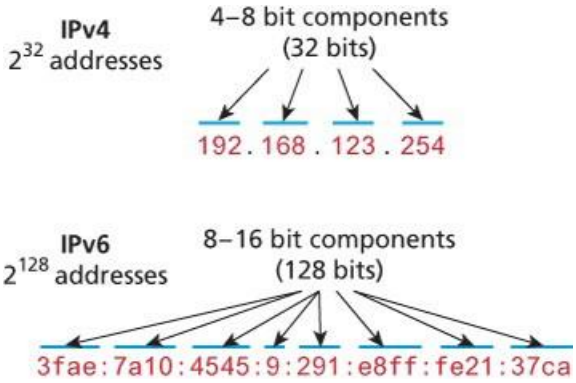
The Internet uses the **Internet Protocol (IP) addresses**, which are **numeric codes** that uniquely identify destinations on the Internet. As can be seen in following figure, every device connected to the Internet has such an IP address.



IP-Address (MCQ + SQ)

There are **two types of IP** addresses: **IPv4 and IPv6**. IPv4 addresses are the IP addresses from the original TCP/IP protocol. In IPv4, **12 numbers** are used (implemented as **four 8-bit**

integers), written with a **dot** between each integer (see following figure). Since an **unsigned 8-bit integer's maximum value is 255**, four integers together can encode approximately 4.2 billion unique IP addresses.

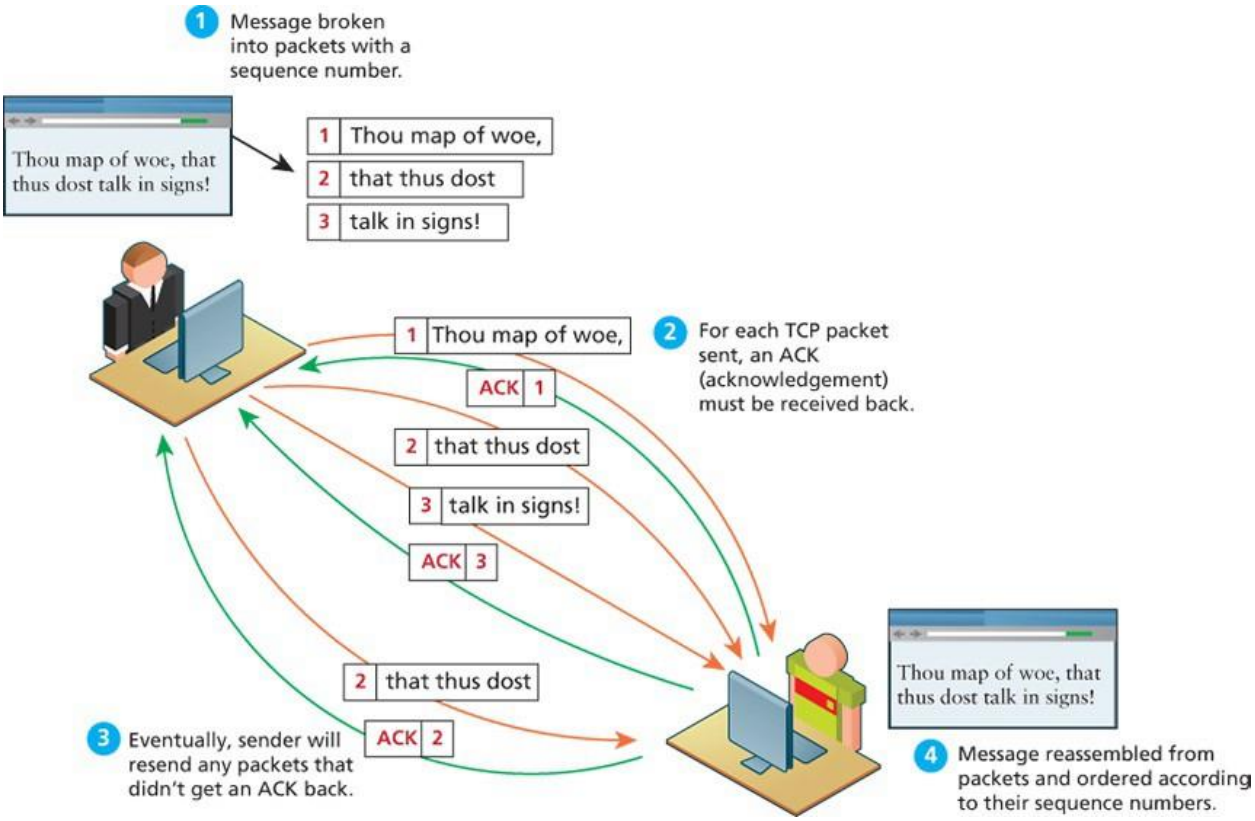


The decision to make IP addresses 32 bits limited the number of hosts to 4.2 billion. As more and more devices connected to the Internet the supply of addresses dwindled, especially in some local areas that had already distributed their allotment.

To future-proof the Internet against the 4.2 billion limits, a new version of the IP protocol was created**, IPv6**. This newer version uses **eight 16-bit** integers for **2128 unique addresses**, over a billion billion times the number in IPv4. **These 16-bit integers are normally written in hexadecimal, due to their longer length.**

<u>Transport Layer</u> (MCQ + SQ)

The transport layer ensures transmissions arrive in order and without error. This is accomplished through a few mechanisms. First, the data is broken into packets formatted according to the **Transmission Control Protocol (TCP)**. Each data packet has a header that includes a sequence number, so the receiver can put the original message back in order, no matter when they arrive. Secondly, each packet acknowledges its successful arrival back to the sender so in the event of a lost packet, the transmitter will realize a packet has been lost since no ACK arrived for that packet. That packet is retransmitted, and although out of order, is reordered at the destination, as shown in following figure. This means you have a guarantee that messages sent will arrive and will be in order.



Sometimes we do not want guaranteed transmission of packets. Consider a live multicast of a soccer game, for example. Millions of subscribers may be streaming the game, and the broadcaster can't afford to track and retransmit every lost packet. A small loss of data in the feed is acceptable, and the customers will still see the game. An Internet protocol called **User Datagram Protocol (UDP)** is used in these scenarios in lieu of TCP. Other examples of UDP services include Voice Over IP, many online games, and Domain Name System (DNS).

Application Layer (MCQ + SQ)

Application layer protocols implement process-to-process communication and are at a higher level of abstraction in comparison to the low-level packet and IP address protocols in the layers below it. There are many application layer protocols. A few that are useful to web developers include the following:

HTTP. The Hypertext Transfer Protocol is used for web communication. **(MCQ)**

SSH. The **Secure Shell Protocol** allows remote command-line connections to servers. **(MCQ)**

FTP. The **File Transfer Protocol** is used for transferring files between computers.

POP/IMAP/SMTP. Email-related protocols for transferring and storing email. **(MCQ)**

DNS. The **Domain Name System** protocol used for resolving domain names to IP addresses. **(MCQ)**

Textbook

"Fundamentals of Web Development", 2nd Edition by Randy Connolly and Ricardo Hoar, 2018, Pearson

Credits

Reading Material and figures are provided by Pearson Education.

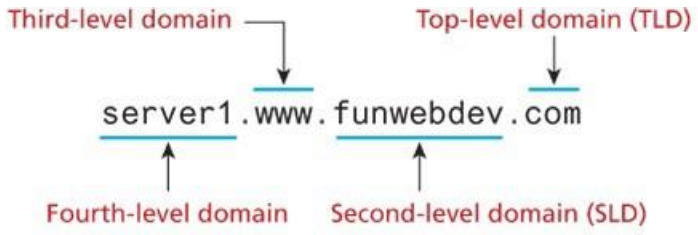
Module 09:

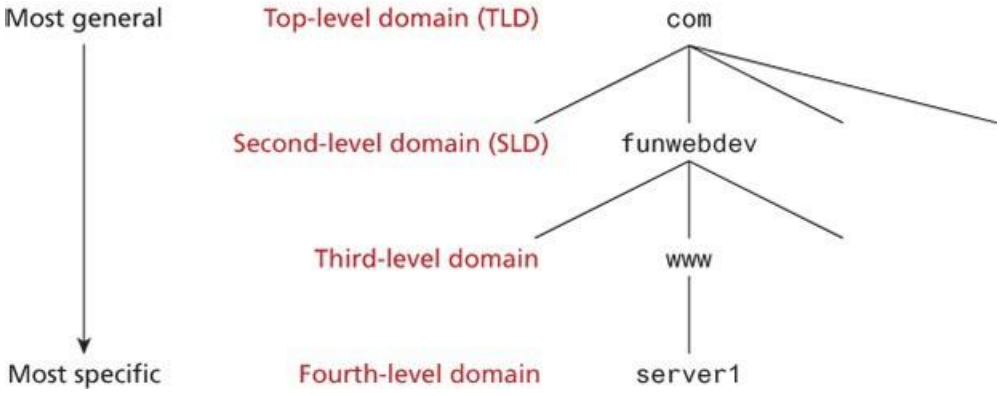
Domain Name System (SQ)

DNS is one of the core systems that make an easy-to-use Internet possible (DNS is used for email as well). One can imagine how unpleasant the Internet would be if you had to remember IP addresses instead of names. Rather than google.com, you'd have to type 216.58.216.78.

Name levels in a domain (MCQ)

A domain name can be broken down into several parts. They represent a hierarchy, with the rightmost parts being closest to the root at the "top" of the Internet naming hierarchy. All domain names have at least a **top-level domain (TLD)** name and a **second-level domain (SLD)** name. Most websites also maintain a third-level WWW subdomain and perhaps others. Following figure illustrates a domain with four levels.





Types of top-level domains

The rightmost portion of the domain name (to the right of the rightmost period) is called the top-level domain. For the top level of a domain, we are limited to **two broad categories**, plus **a third** reserved for other use. They are:

Generic top-level domain (gTLD) (MCQ + SQ)

Unrestricted. TLDs include .com, .net, .org, and .info. **(MCQ)**

Sponsored. TLDs including .gov, .mil, .edu, and others. These domains can have requirements for ownership and thus new second-level domains must have permission from the sponsor before acquiring a new address.

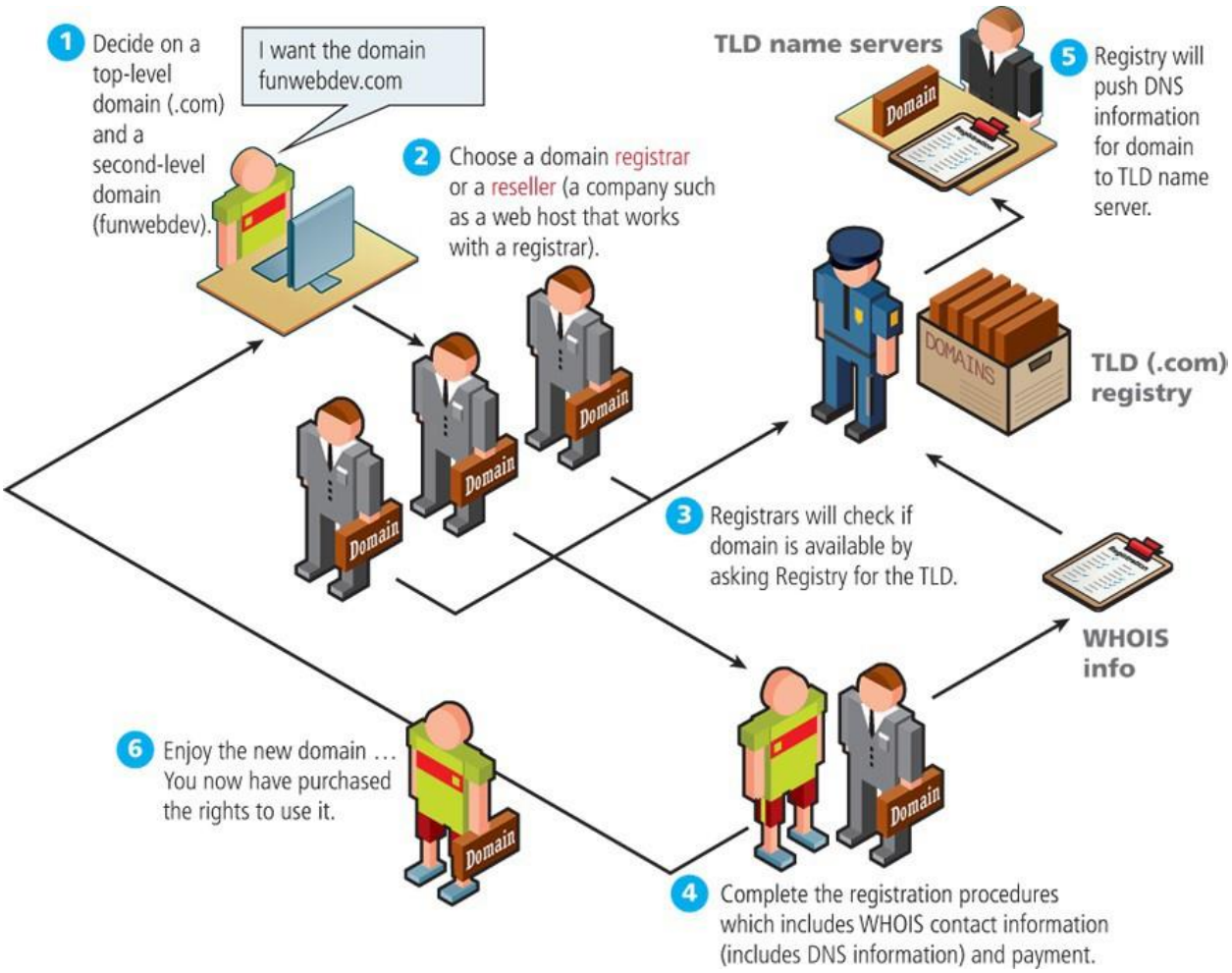
New. From January to May of 2012, companies and individuals could submit applications for new TLDs. TLD application results were announced in June 2012, and include a wide range of both contested and single applicant domains. These include corporate ones like .apple, .google, and .macdonalds, and contested ones like .buy, .news, and .music.5

Country code top-level domain (ccTLD) (MCQ)

- TLDs include .us, .ca, .uk, and .au. At the time of writing, there were **252 codes registered**.6 These codes are under the control of the countries which they represent, which is why each is administered differently. In the **United Kingdom**, for example, commercial entities and businesses must register subdomains to **co.uk** rather than second-level domains directly. In **Canada, .ca** domains can be obtained by any person, company, or organization living or doing business in Canada. Other countries have peculiar extensions with commercial viability (such as **.tv for Tuvalu**) and have begun allowing unrestricted use to generate revenue.
- Since some nations use nonwestern characters in their native languages, the concept of the internationalized top-level domain name (IDN) has also been tested with great success in recent years. Some IDNs include Greek, Japanese, and Arabic domains (among others) which have test domains at http://παράδειγμα.δοιμή, http:// 例え. テスト, and http:// **مثال**. **إختبار**, respectively.

Domain registration

Special organizations or companies called domain name registrars manage the registration of domain names. These domain name registrars are given permission to do so by the appropriate generic top-level domain (gTLD) registry and/or a country code top-level domain (ccTLD) registry. In the 1990s, a single company (Network Solutions Inc.) handled the com, net, and org registries. By 1999, the name registration system changed to a market system in which multiple companies could compete in the domain name registration business. A single organization—the nonprofit **Internet Corporation for Assigned Names and Numbers** (ICANN)—still oversees the management of top-level domains, accredits registrars, and coordinates other aspects of DNS. At the time of writing this chapter, there were almost 1000 different ICANN-accredited registrars worldwide. Following figure illustrates the process involved in registering a domain name.



The process involves six steps shown as follows:

1. "Decide on a top-level domain (.com) and a second level domain (funwebdev)." The illustration shows a user at a server who says, "I want the domain funwebdev.com"

2. "Choose a domain registrar or a reseller (a company such as a web host that works with a registrar." The illustration shows three formally dressed figures carrying suitcases labeled as "Domain."

3. "Registrars will check if domain is available by asking Registry for TLD." The illustration points to a security guard standing next to a box full of domain slabs, labeled "TLD (.com) registry."

4. "Complete the registration procedures which includes WHOIS contact information (includes DNS information) and payment." The illustration shows the user talking to a domain registrar. An arrow points to a paper labeled "WHOIS info" which points to the security guard standing next to "TLD (.com) registry."

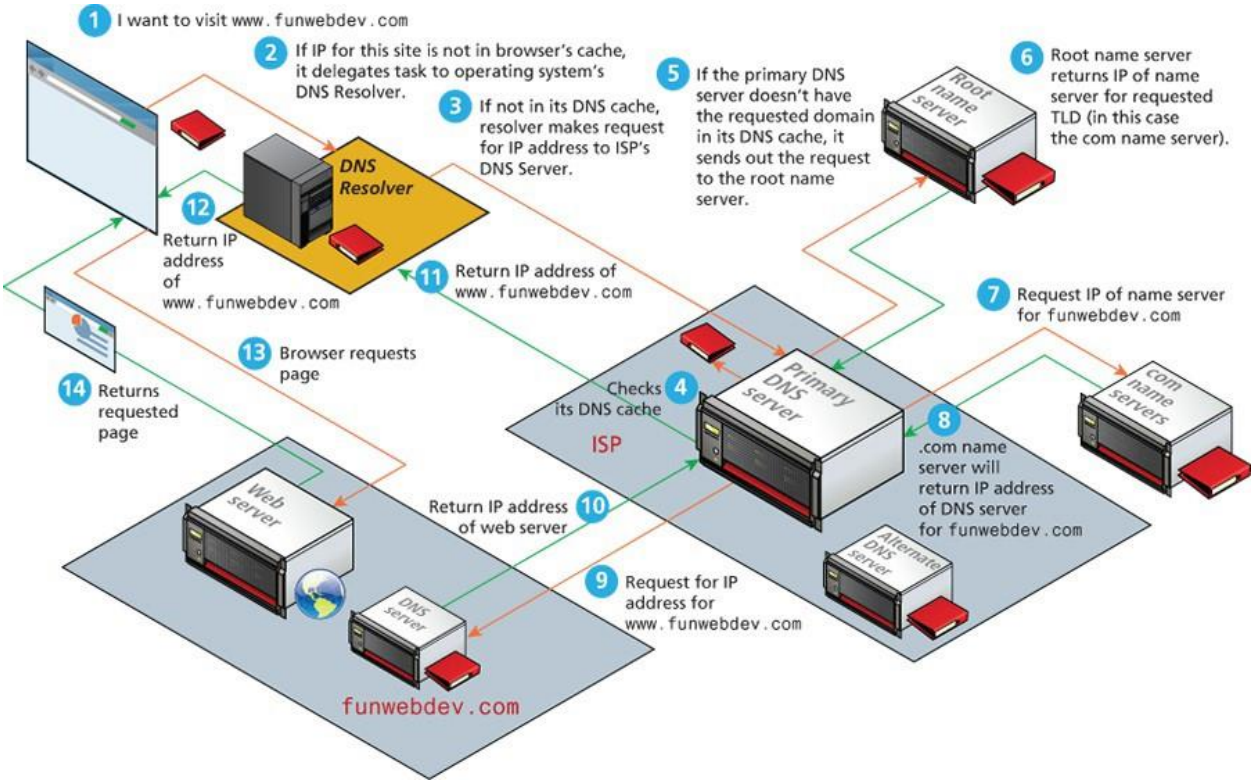
5. “Registry will push DNS information for domain to TLD name server.” The illustration shows a formally dressed figure at a desk labeled “TLD name servers”, with a domain and registration form.

6. “Enjoy the new domain...You now have purchased the rights to use it.”

The illustration shows the user holding a suitcase labeled as “Domain”.

Address Resolution

While domain names are certainly an easier way for users to reference a website, eventually your browser needs to know the IP address of the website in order to request any resources from it. **DNS provides a mechanism for software to discover this numeric IP address. This process is referred to as address resolution**. When you request a domain name, a computer called a domain name server will return the IP address for that domain. With that IP address, the browser can then make a request for a resource from the web server for that domain. What actually happens during address resolution is more complicated, as can be seen in following figure.



1. The resolution process starts at the user's computer. When the URL www.funwebdev.com is requested (perhaps by clicking a link or typing it in), the browser will begin by seeing if it already has the IP address for the domain in its cache. If it does, it can jump to step in the diagram.

2. If the browser doesn't know the IP address for the requested site, it will delegate the task to the DNS resolver, a software agent that is part of the operating system. The DNS resolver also keeps

a cache of frequently requested domains; if the requested domain is in its cache, then the process jumps to step.

3. Otherwise, it must ask for outside help, which in this case is a nearby DNS server, a special server that processes DNS requests. This might be a computer at your Internet service provider (ISP) or at your university or corporate IT department. The address of this local DNS server is usually stored in the network settings of your computer's operating system. This server keeps a more substantial cache of domain name/IP address pairs. If the requested domain is in its cache, then the process jumps to step.

4. If the local DNS server doesn't have the IP address for the domain in its cache, then it must ask other DNS servers for the answer. Thankfully, the domain system has a great deal of redundancy built into it. This means that in general there are many servers that have the answers for any given DNS request. This redundancy exists not only at the local level but at the global level as well.

5. If the local DNS server cannot find the answer to the request from an alternate DNS server, then it must get it from the appropriate top-level domain (TLD) name server. For funwebdev.com this is .com. Our local DNS server might already have a list of the addresses of the appropriate TLD name servers in its cache. In such a case, the process can jump to step.

6. If the local DNS server does not already know the address of the requested TLD server (for instance, when the local DNS server is first starting up it won't have this information), then it must ask a root name server for that information. The DNS root name servers store the addresses of TLD name servers. **IANA (Internet Assigned Numbers Authority)** authorizes 13 root servers, so all root requests will go to one

of these 13 roots. In practice, these 13 machines are mirrored and distributed around the world (see http://www.root-servers.org/ for an interactive illustration of the current root servers); at the time of writing, there are over 500 root server machines. With the creation of new commercial top-level domains in 2012, approximately 2000 or so new TLDs has come online, creating a heavier load on these root name servers.

7. After receiving the address of the TLD name server for the requested

domain, the local DNS server can now ask the TLD name server for the

address of the requested domain. As part of the domain registration

process, the address of the domain's DNS servers are sent to the TLD name servers, so this is the information that is returned to the local DNS server in step.

8. The user's local DNS server can now ask the DNS server (also called a second-level name server) for the requested domain (www.funwebdev.com); it should receive the correct IP address of the web server for that domain. This address will be stored in its own cache so that future requests for this domain will be speedier. That IP address can finally be returned to the DNS resolver in the requesting computer,

as shown in step.

9. The browser will eventually receive the correct IP address for the requested domain, as shown in step. Note: If the local DNS server were unable to find the IP address, it would return a failed response, which in turn would cause the browser to display an error message.

10. Now that it knows the desired IP address, the browser can finally send out the request to the web server, which should result in the web server responding with the requested resource.

Textbook

"Fundamentals of Web Development", 2nd Edition by Randy Connolly and Ricardo Hoar, 2018, Pearson

Credits

Reading Material and figures are provided by Pearson Education.

Module 10:

Topic 10: Uniform Resource Locators (MCQ)

Uniform Resource Locators (URL) play an important role in World Wide Web. URL has different components. They components are protocol, domain, path, query string and fragment. The functions of these parts are explained below.

10.1 Protocol

This part describes the protocol through which a web-page is accessed. When we open a website, we need to specify the protocol first. **Most of the unsecured websites are accessed through HTTP protocol while in the case of secured websites, HTTPS protocol is used.** Web browsers also support other protocols. For example, FTP protocol is used for accessing files. In figure 1, the protocol used is HTTP.

10.2 Domain

The address of a website is called domain. Domain consists of top-level domain, country code top-level domain, second level domain, third level domain etc. Domain name Server finds IP address against this name and this website can be accessed using that IP address. In figure 1, www.funwebdev.com is domain.

10.3 Port (MCQ)

It is an optional part of a URL. Usually, it is not visible. Most of the times**, HTTP servers listen at port 80.** A colon is added after domain name and then integer port number is specified.

10.4 Path

After domain, the path, of the file we want to access, is specified. In figure 1, index.php is path of a file being accessed. When path is not mentioned, default files like index.html or index.php are returned.

10.5 Query String (MCQ)

It is an information passed to the file. Query strings are mostly used in dynamic websites. Information in query string is sent to server in the form of key value pairs**. It starts from delimiter '?',** and if there are more than one keys in a query string, they are **separated using '&'.** In figure 1, query string is page=17.

10.6 Fragment (MCQ)

Fragment is used to request a specific portion of a web-page. It is represented using **'#'symbol**. In figure 1, article is a fragment, and browser automatically scrolls the website to it.



Figure-1: Uniform Resource Locators (taken from Fundamentals of Web Development second edition 2017)

Module 11:

Topic 11: Hypertext Transfer Protocol

Hypertext Transfer Protocol (HTTP) is the backbone of World Wide Web (WWW) which is used to access and visit websites. Headers are used in HTTP. A sample illustration of HTTP is given in figure 2. When a client requests to access a web-page, a piece of information is sent to server called request header. It has different components like methods (GET, POST), the path a client wants to visit, and the file a client wants to access on a server. These headers are automatically generated by a browser, and sent to server side. It can also contain some additional information e.g. version of protocol, host, and IP address. If multiple websites are hosted on a single web server, there IP address will remain the same. Another important part of HTTP is user agent which contains information about client's operating system and browser. A server can generate dynamic contents on the basis of this information.

When a web server receives a request, it sends response header and content against it. Response header can contain response code, date & time, information about server, and content length etc.

When a client fills a form, its data is sent to server suing POST method. The same can be done by using GET method, but it uses query string to send information to server. The comparison of POST and GET is shown in figure 3.

Different response codes are used to determine how a server handled a client's request**. 200 codes are for successful responses, 300 are for redirection-related responses, 400 codes are client errors and 500 codes are server errors.** For example, some response codes and their purposes are given below.

(MCQ V.imp)

200: OK

301: Moved Permanently

400: Bad Request

404: Not found

500: Internal server error

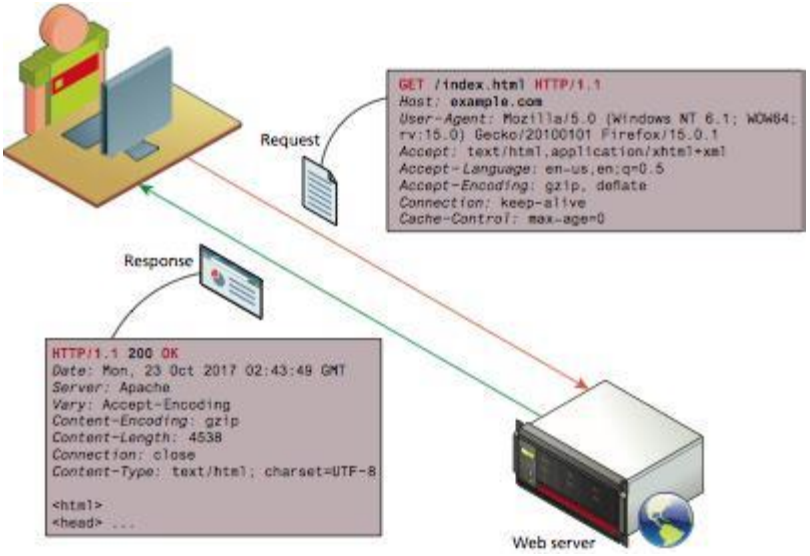


Figure-2: HTTP (taken from Fundamentals of Web Development second edition 2017)

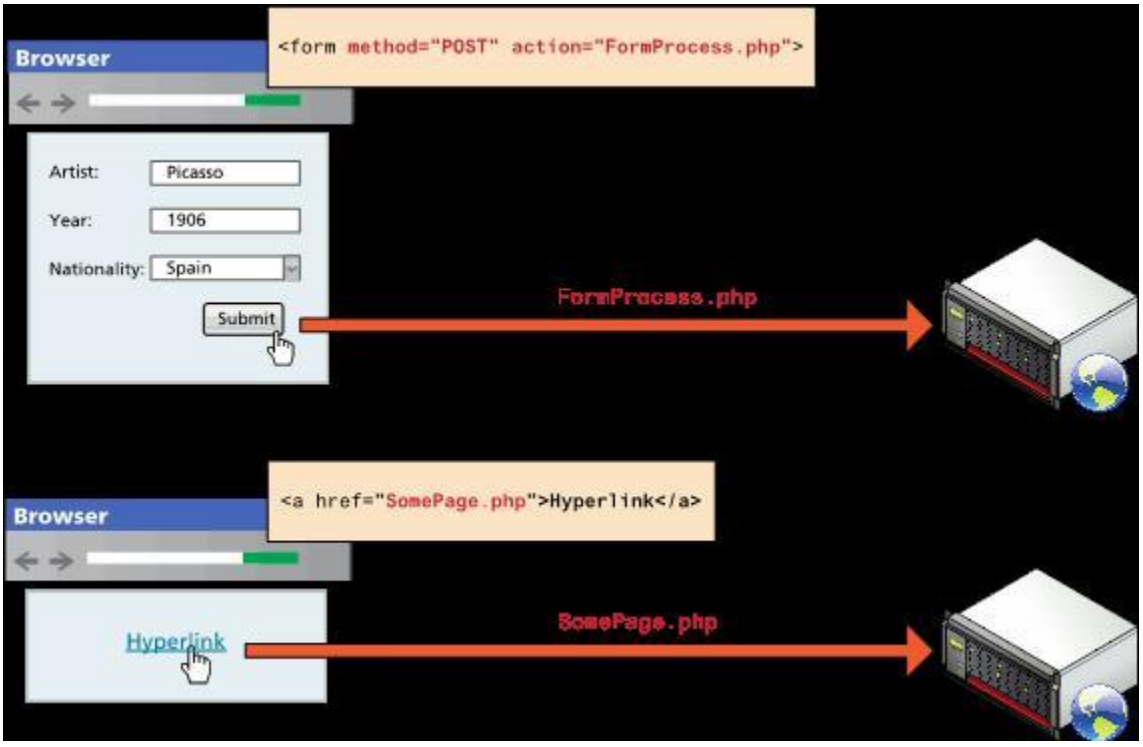


Figure-3: POST Vs GET (taken from Fundamentals of Web Development second edition 2017)

Module 12:

How a web browser works? (SQ)

When we open a website, many processes run at backend. When we request for a webpage, we get its HTML contents first, which are in textual form. If that webpage contains any images, videos, styles sheets or scripts, all of these external files are fetched one by one. Browser requests and receives these files from server. One request may contain further multiple requests to get webpage content. When browser gets these contents, it renders and displays our page by using them.

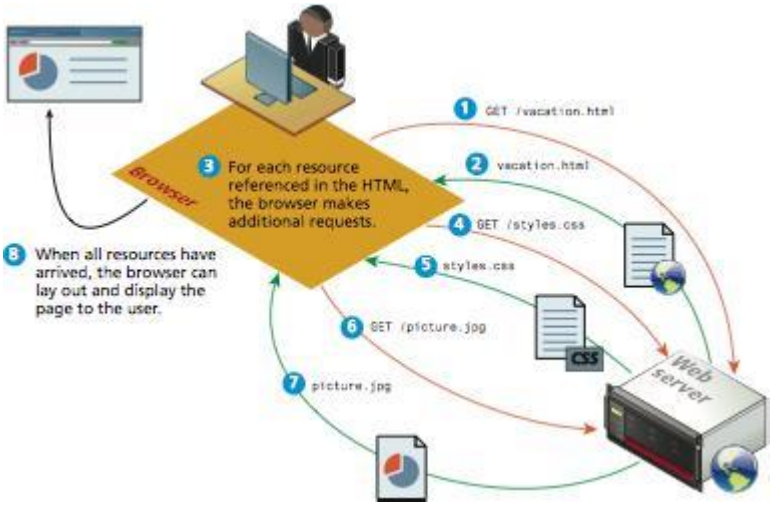
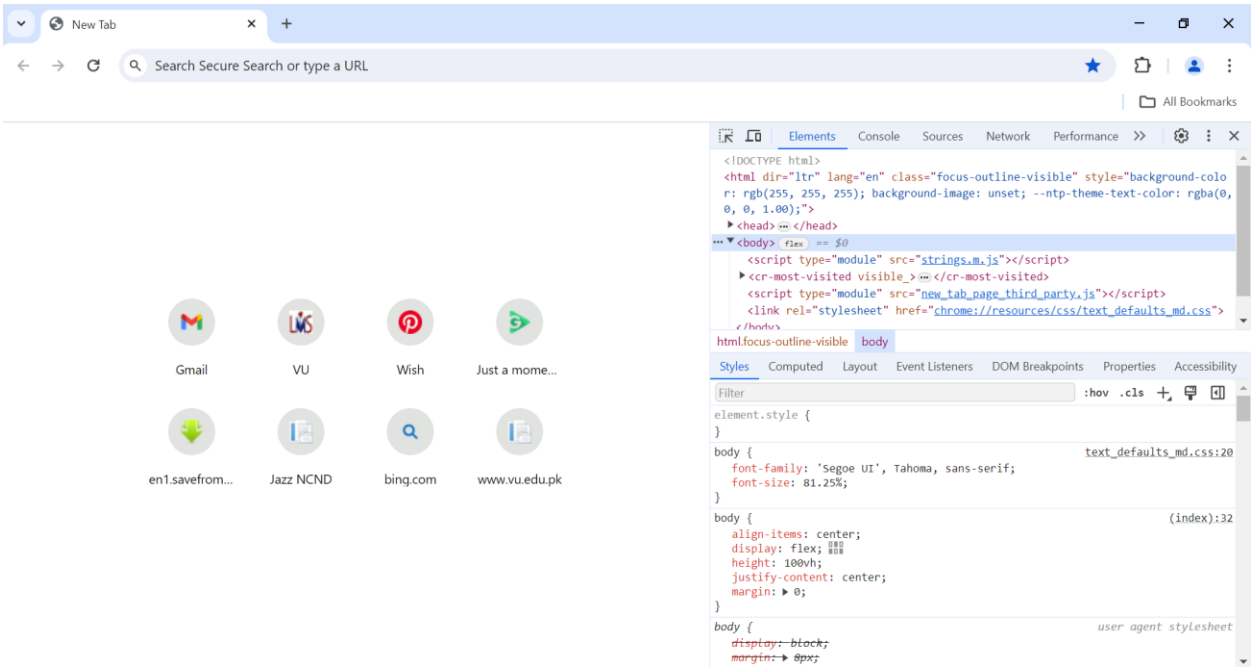


Figure-4: Web browser working (taken from Fundamentals of Web Development second edition 2017)

12.1 Load times and cascades

We can check the requests and responses against them, and time taken for responses, when we open a website. It can be done using developer tools in Chrome/Firefox. This tool is particularly intended for developers and thus called a developer tool. For example, in Chrome, we can open developer tools using browser menu **(Ctrl+Shift+I** shortcut can also be used). These developer tools contain different components for different purposes. For example, the component 'Network' contains information related to network.



12.2 Browser Rendering

When a browser receives all the contents of a webpage, it renders and then displays it. Different browsers, for this purpose, use different procedures according to their standards which have been devised by **World Wide Web Consortium (W3C)**. Therefore, a web developer should test a website using different browsers.

12.3 Browser Caching

When we visit a website and fetch some images and files from it, they are stored in browser cache. Next time, when we need that file from the same server, the file will be retrieved from browser cache instead of server, resulting in fast browsing. Caching can also be disabled to avoid caching, like done in banking websites.

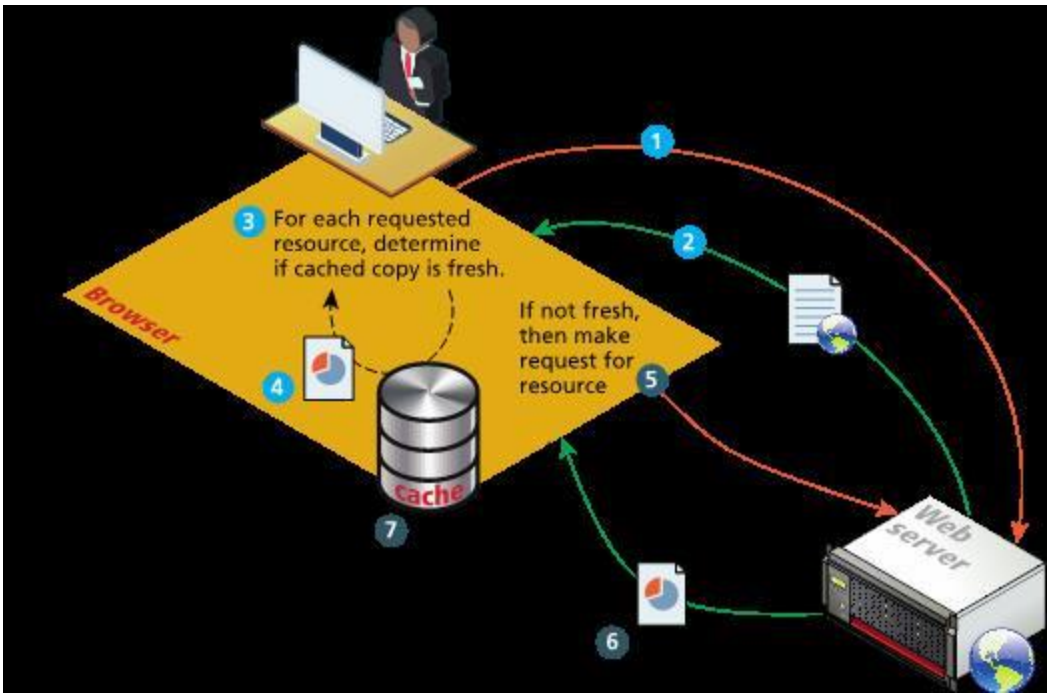


Figure-5: Browser Caching (taken from Fundamentals of Web Development second edition 2017)

12.4 Browser Features (MCQ + SQ)

Browsers contain certain features to facilitate users. Some features and their purposes are given below.

Search Engine Integration: Suggests search query in search bar. **(MCQ)**

URL Auto-completion: Gives suggestions on the basis of address bar input. **(MCQ)**

Form Auto-completion: Browsers saves some values and fills them automatically in upcoming forms. **(MCQ)**

Cloud sync of user data: Search data becomes independent of device and can be available at any device against specific login credentials. **(MCQ)**

Phishing website detection: Some browsers also identify websites which can hack user's data. **(MCQ)**

Secure connection visualization: A lock symbol is shown to represent secure connection. **(MCQ)**

12.5 Browser Extensions

Extensions can modify the page being rendered. They are designed for specific purposes. For example, extensions like ‘Lighthouse’ and ‘React developer tools’ are very useful for developers while extensions like ‘Adblock’ and ‘Third Party Plugins’ are used by general public.

Module 13:

Introduction to web servers (SQ: types of servers)

It is also a computer but more powerful than the computers used at homes. Web servers follow a certain application stack. This application stack consists of operating system, web server software, a database, and scripting language for dynamic requests.

13.1 Application Stacks (MCQ)

The most commonly used application stack is LAMP which contains **L**inux operating system, **A**pache web server, **M**ySQL database, and **P**HP scripting language. It is traditionally used for dynamic websites. There are other stacks as well. For example, **WAMP, WISA, and MEAN**. Therefore, a full-stack developer is supposed to have expertise of working on operating system, web server, frontend development, and backend development.

13.2 Operating Systems

Mostly used operating systems are Linux, Mac, and Windows.

13.3 Web server software

Some popular web server software are Apache, Nginx, and IIS.

13.4 Database software

Relational databases like MySQL, PostgreSQL, and Sqlite are mostly used. However, document based databases are also being used these days. One important example of document based databases is MongoDB.

13.5 Scripting languages

The content of a dynamic website can be generated in almost every scripting language. However, PHP is world-wide used for this purpose. These days, ASP.NET, Python, and Node.js are also being used as scripting languages.

Module 14:

Architecture of web-based system

14.1 Multi-tier architecture
Web development can be realized at multiple tiers.

2-tier architecture

When we access a website, at least 2-tiers are created i.e. Client-side tier and Server-side tier. However, in the perspective of web development, it becomes easy if we divide it into more tiers. It also eases the maintenance of a website.

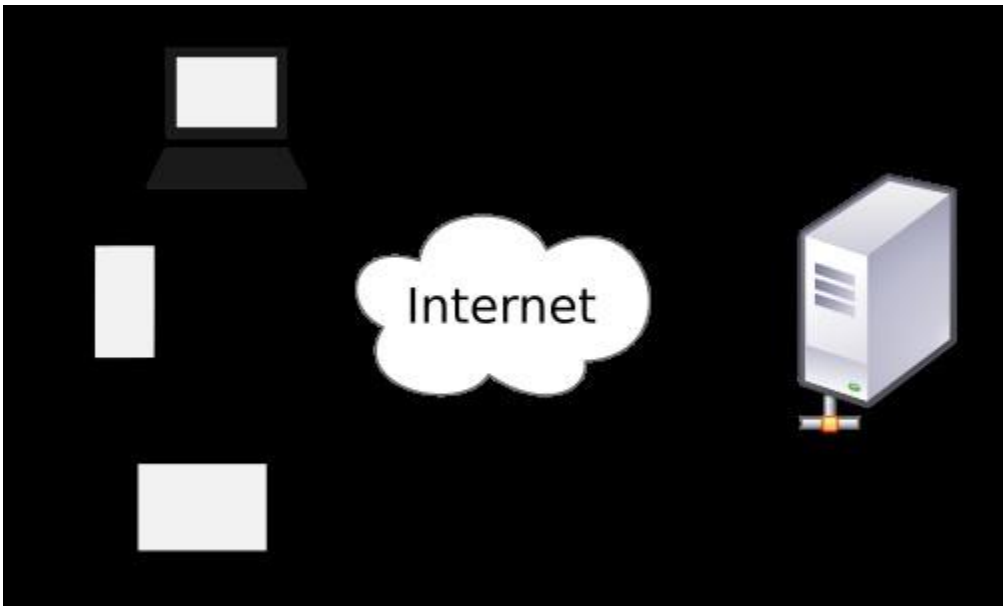


Figure-6: 2-tier architecture (taken from https://en.wikipedia.org/wiki/File:Client-server-model.svg)

3-tier architecture (SQ)
In software engineering perspective, these tiers can be broken down to further tiers. In 3-tier architecture, **first tier is responsible for content display** (Presentation Layer), **second tier is business logic for generating dynamic content, third tier is database layer which stores data and manages database**. The advantage of separating presentation and business logic layers is that when we change business logic for some users, we do not need to change presentation layer necessarily.

Module 15:

Introduction to HTML

HTML stands for hypertext markup language. Markup is a way to indicate information about the content that is distinct from the content. There has been different version of HTML, the **latest is HTML5**. To understand markup, think of an assignment you submitted to your teacher. Teacher checks it and marks it for what can be improved or which mistakes student made which can be corrected. The feedback of teacher can be thought as markup. To make distinction between assignment and markup, teacher may use different color pen.

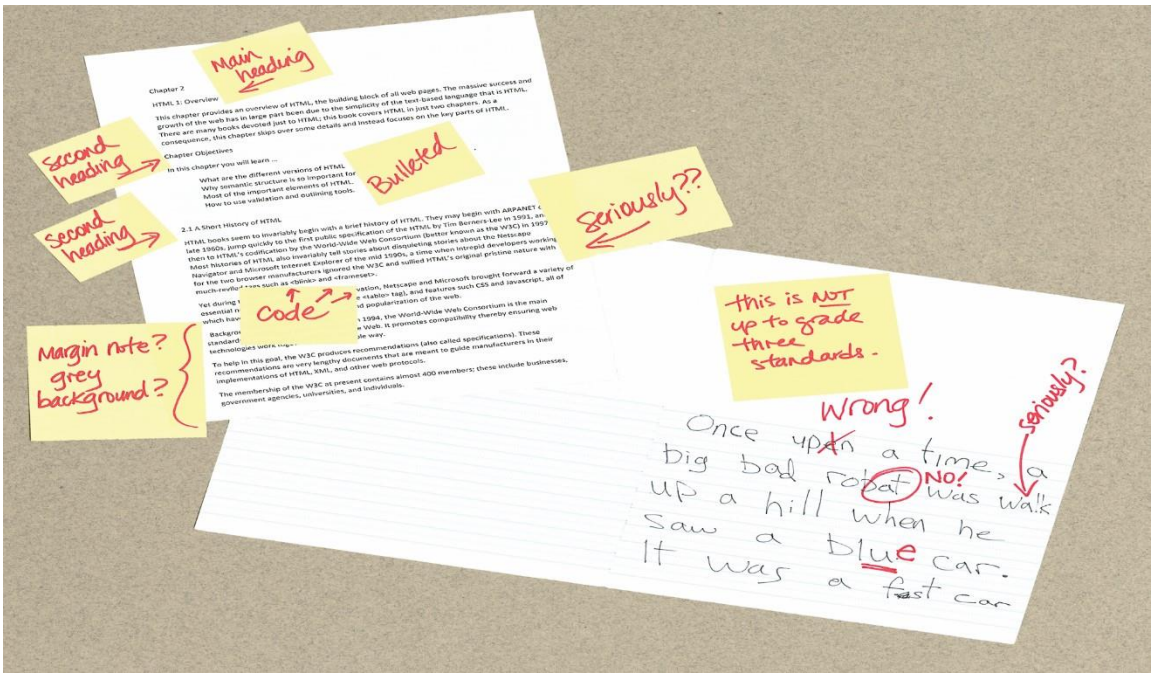


Figure-7: Markup (taken from Fundamentals of Web Development second edition 2017)

HTML validator, a tool of W3C, is used to identify errors in an HTML page and give recommendations in the form of warnings.

15.1 HTML5

It is the **latest version of HTML** supported by all of the modern browsers.

15.2 Elements and attributes

An HTML document contains HTML elements, also called **HTML tags**. Every HTML element has a unique name which can be written after the angle bracket (<). HTML elements can also contain attributes and the content within the tag. The figure given below shows the syntax of anchor tag, an attribute of link, and content. Most of tags come in pair, means they have closing tags as well.



Figure-8: Anchor Tag(taken from Fundamentals of Web Development second edition 2017)

15.3 An empty element

An empty element does not contain any text content; instead, it is an instruction to the browser to do something.

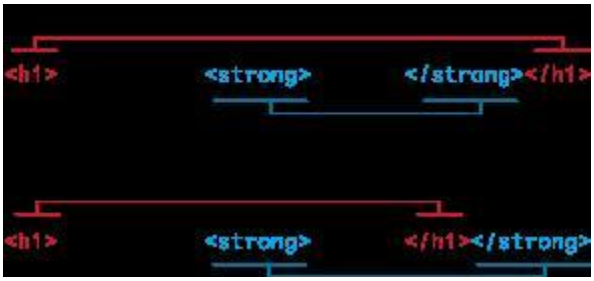
In XHTML, empty elements had to be terminated by a trailing slash. However, the trailing slash in empty elements is optional in HTML5.



Figure-9: Empty Element(taken from Fundamentals of Web Development second edition 2017)

15.4 Nesting element

Developers must be careful while using nesting elements. While closing, the nested element must be closed first. The example of correct and incorrect nesting is shown below in figure 10. In figure 11, the relationship of nested tag is shown. Nested elements can also be shown in the form of tree containing roots and children (figure 12).



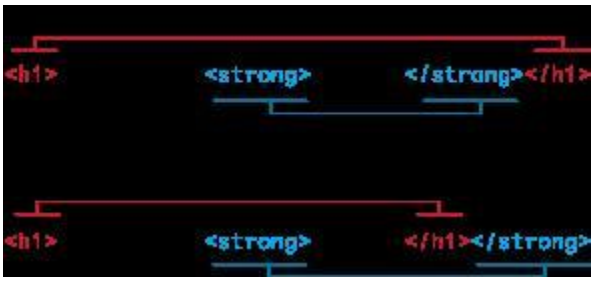


Figure-10: Correct and incorrect Nesting (taken from Fundamentals of Web Development second edition 2017)

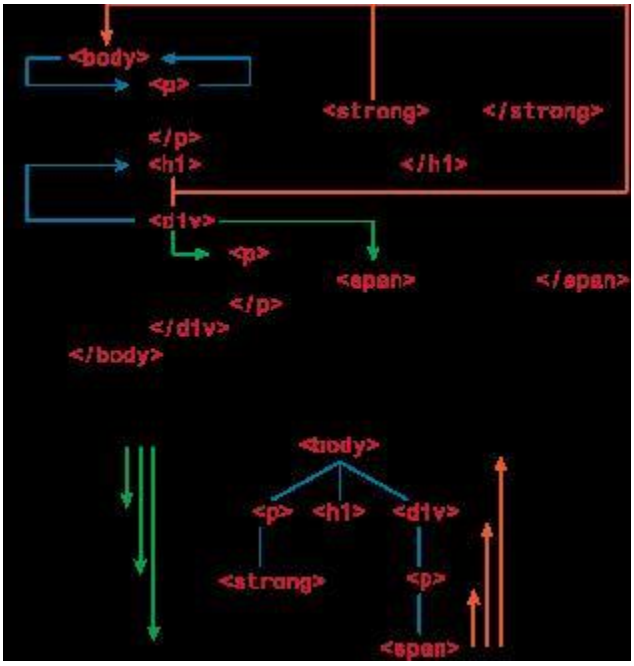


Figure-11: Relationship between tags (taken from Fundamentals of Web Development second edition 2017)

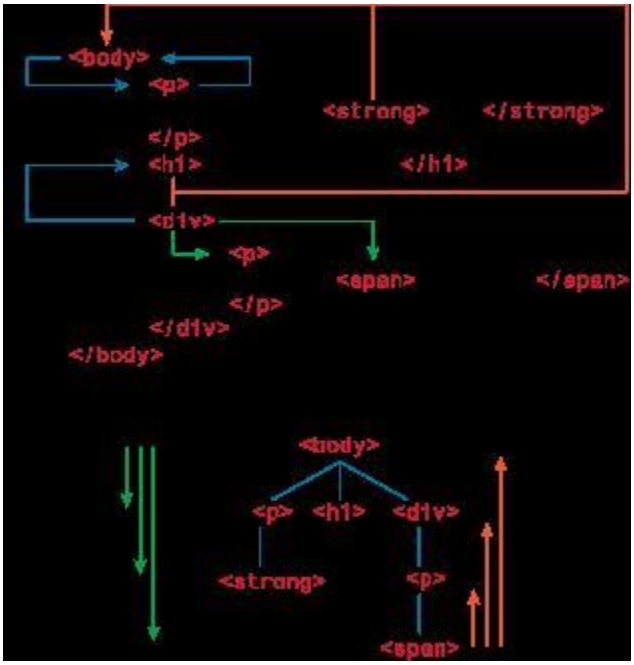


Figure-12: Tree structure of relationships (taken from Fundamentals of Web Development second edition 2017)

Module 16:

[**for you information:** from now to onward only boldness showing important quiz and sometime short question or long, specific abbreviations are not used]

Common HTML tags; Heading, Paragraphs, Divs; Inline Text Elements; Images; HTML Character Entities

HTML is only used for structuring and storing content. It is not used for styling but style sheets can be used for this purpose. Common HTML tags are explained below.

16.1 DOCTYPE (SQ+MCQ)
This tag is used to inform web server and browser that the document is an HTML document. It is written in the start of document. Its syntax is **<!DOCTYPE html>.**

16.2 Head and Body tags (LQ)

External files, document title, and meta data about document are mentioned in the 'head' tag. **The contents of document are mentioned in the 'body' tag.** Earlier, parent 'html' tag was necessary but this compulsion has been removed in HTML5. However, most developers still use this tag.

16.3 Meta Tag
This tag contains information about **coding details** of HTML contents.

16.4 Link Tag
This tag is used to **link the external styles sheets** to an HTML document.

16.5 Script Tag
To specify a certain script from an external file, script tag is used.



Figure-13: Head, body, meta, link and script tags (taken from Fundamentals of Web Development second edition 2017)

Heading Tag

HTML provides **six levels of headings (h1 - h6).** Headings are also used by the browser to create a document outline for the page. It is recommended not to use different headings for setting font size.

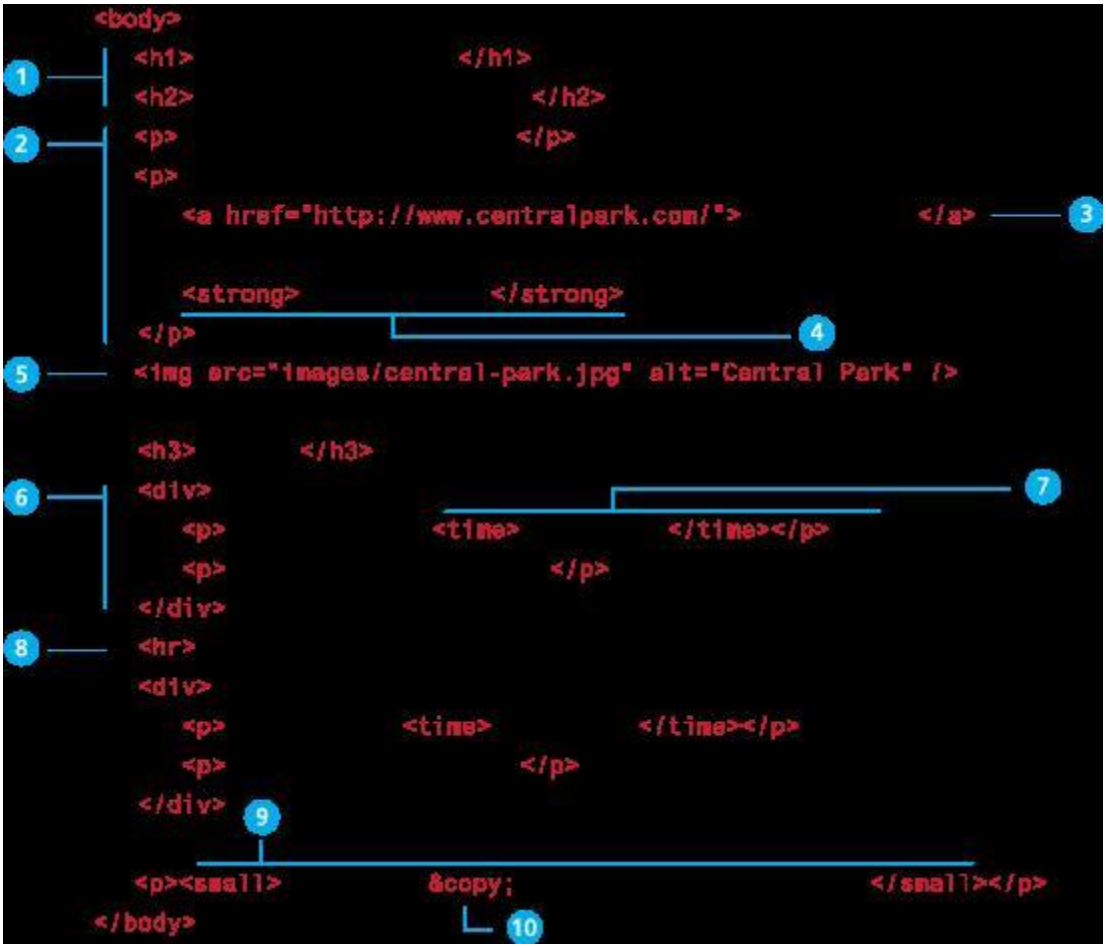


Figure-14: Heading tags (taken from Fundamentals of Web Development second edition 2017)

Paragraphs and Divisions

Paragraph is a container for text and other HTML elements**. <p> tag is used of paragraph**.

Divisions are used to create a logical grouping of content. It is also a container element, and ** tag is used for division.** Following figure demonstrates a paragraph tag.

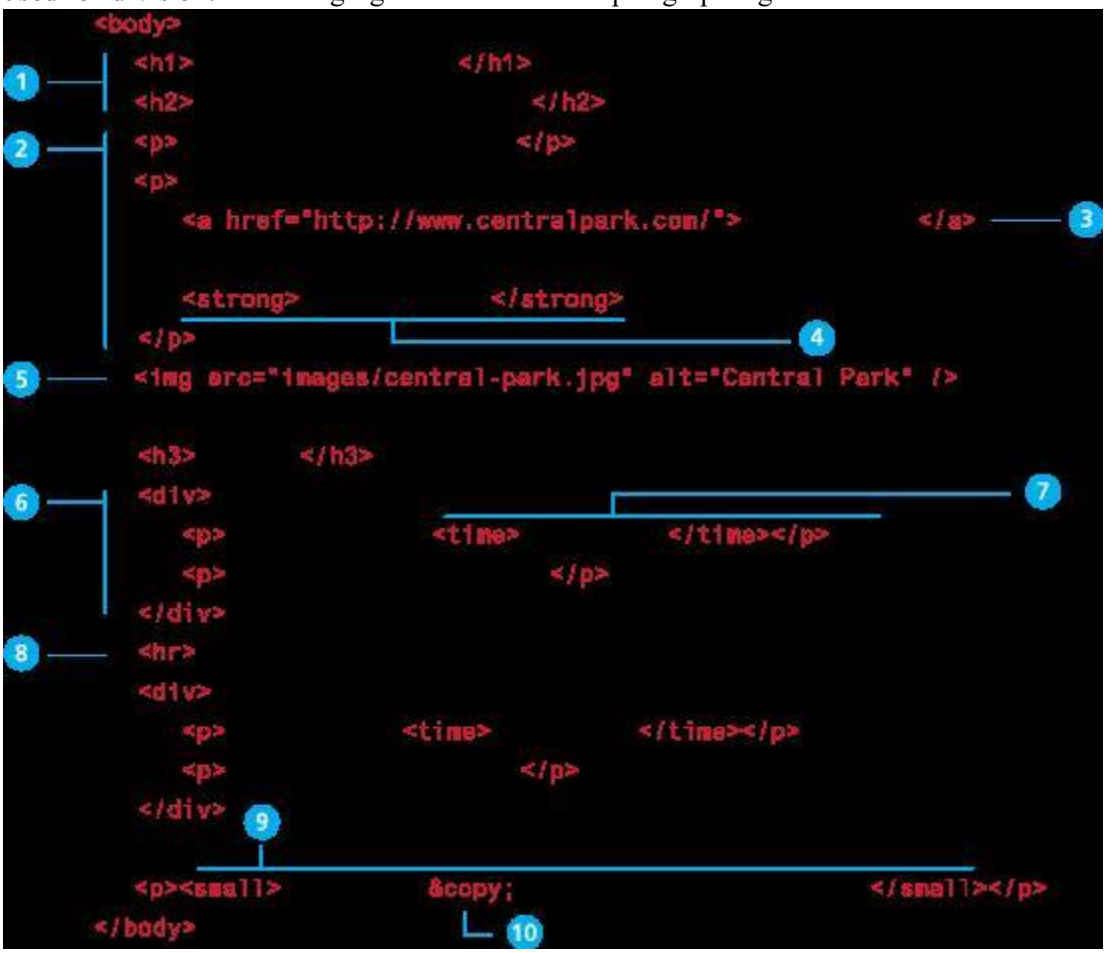


Figure-15: Paragraph tag (taken from Fundamentals of Web Development second edition 2017)

These tags do not create their space separately when they are used in other tags. In other words, these tags do not disrupt the flow. Its examples are <a>, <br>, and <site> etc.

16.9 Block elements

These elements disrupt the existing flow and **take their own space**. Its examples are , <hr>, and <li> etc.

16.10 Images

<img> element is used to display image. It is mandatory to use 'alt' attribute in image tag with the purpose to display text against an image which also helps visually impaired. The figure, given below, explains image tag properly.



Figure-16: Image tag (taken from Fundamentals of Web Development second edition 2017)

16.11 Character entities

These are special characters for symbols for which there is either no easy way to type them via a keyboard or which have a reserved meaning in HTML e.g. **'<', which means that '<' cannot be displayed by just typing '<' because it is a keyword**. Some entities and their description is given below in the table.

Entity	Description
	Nonbreakable space
<	<
>	>
©	©
™	™

Module 17:

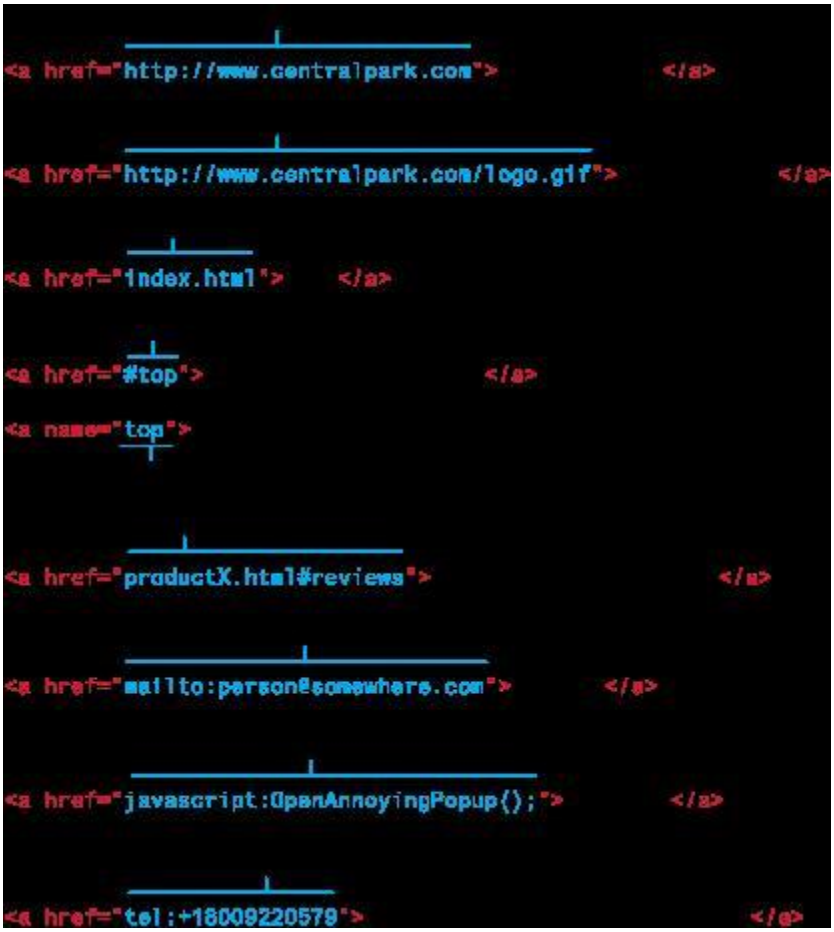
Links

Links are an essential feature of all web pages. Links are created using the **element (the "a" stands for anchor). It has two main parts: the destination and the label.**

As it can be seen in Figure 17, the label of a link can be text or another HTML element such as an image.



Figure-17: Parts of Link (taken from Fundamentals of Web Development second edition 2017)



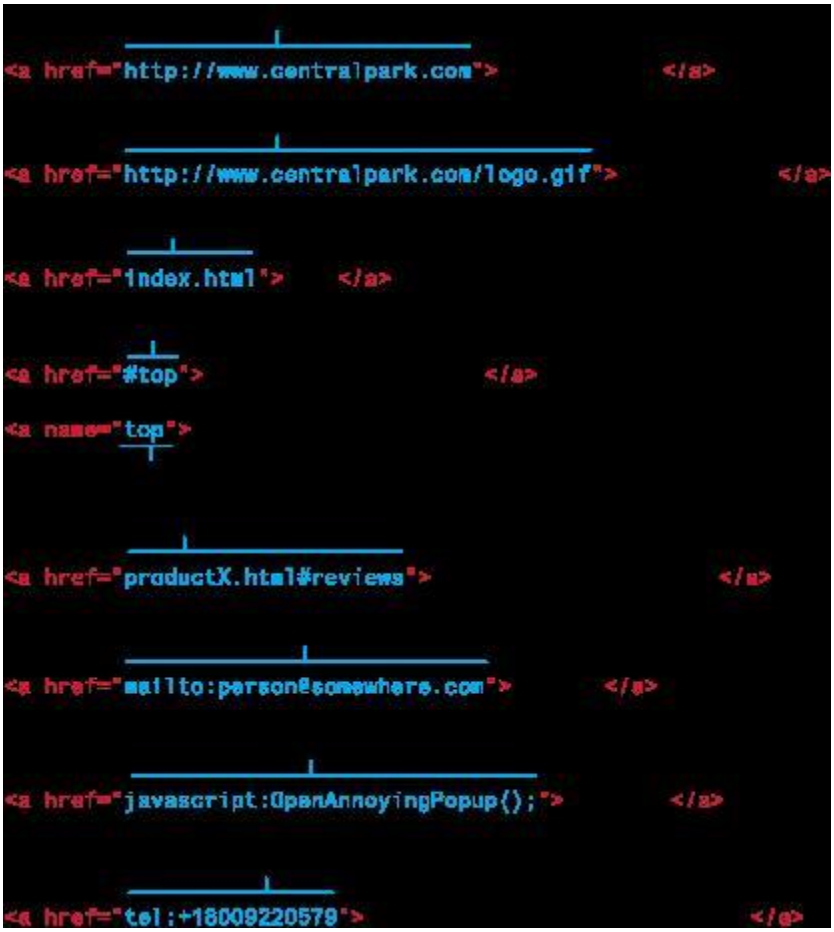


Figure-18: Examples of links (taken from Fundamentals of Web Development second edition 2017)

17.1 Relative Referencing

Whether we are constructing links with the element, referencing images with the <img> element, or including external JavaScript or CSS files, we need to be able to successfully reference files within our site. This requires learning the syntax for so-called relative referencing. Table 17.1 provides examples of the different types of URL referencing.

Relative Link Type	Example
Same directory	<a href="example.html">
Child Directory	<a href="images/logo.gif">
Grandchild/Descendant Directory	<a href="css/images/background.gif">

Parent/Ancestor Directory	<a href="../about.html"> <a href="../../about.html”>
Sibling Directory	<a href="../images/about.html">
Root Reference	<a href="/images/background.gif">

Table 17.1 Relative Referencing

Module 18:

Topic 18: Lists

HTML provides three types of lists which are explained below.

18.1 Unordered lists

Collections of items in no particular order is called unordered list. These are by default rendered by the browser as a bulleted list. However, it is common in CSS to style unordered lists without the bullets. Unordered lists have become the conventional way to markup navigational menus.

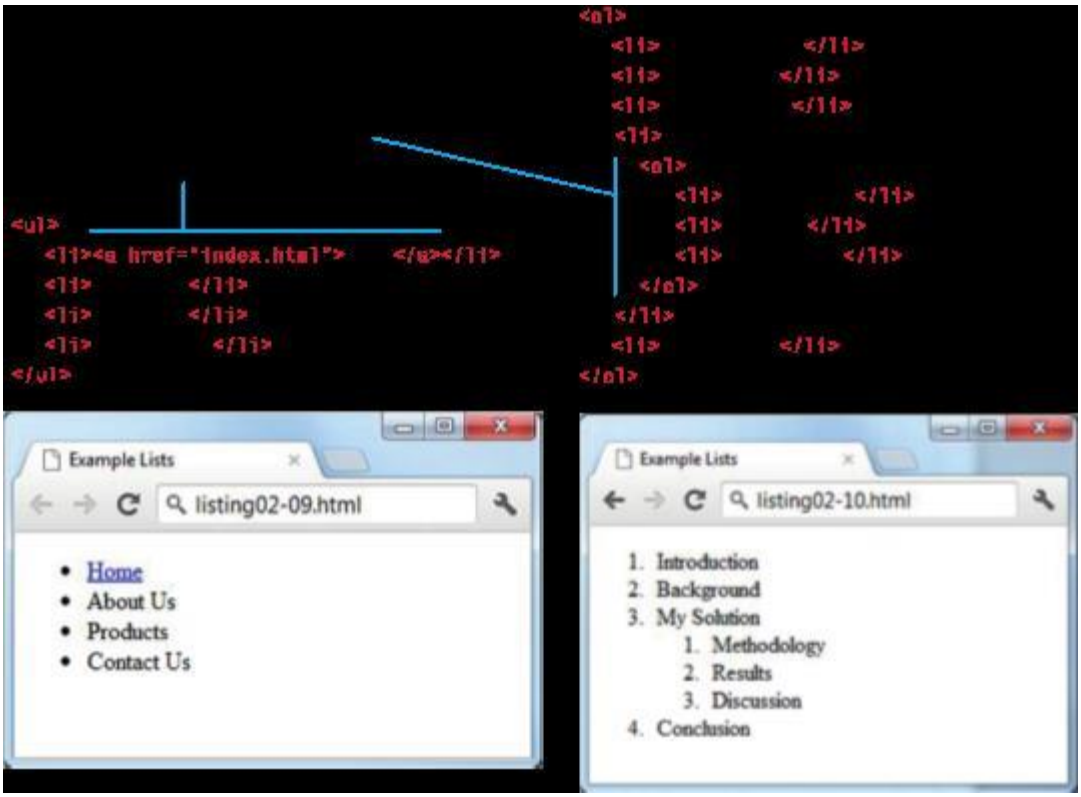


Figure: Unordered lists (taken from Fundamentals of Web Development second edition 2017)

18.2 Ordered lists

Collections of items that have a set order is called ordered list. These are by default rendered by the browser as a numbered list.

```
<!DOCTYPE html>
<ol>
    <li>Introduction</li>
    <li>Background</li>
    <li>My Solution</li>
    <li>Conclusion</li>
</ol>
```

1. Introduction
2. Background
3. My Solution
4. Conclusion

Figure: Ordered lists (taken from Fundamentals of Web Development second edition 2017)

18.3 Nested lists

A list within a list is called nested list, for example:

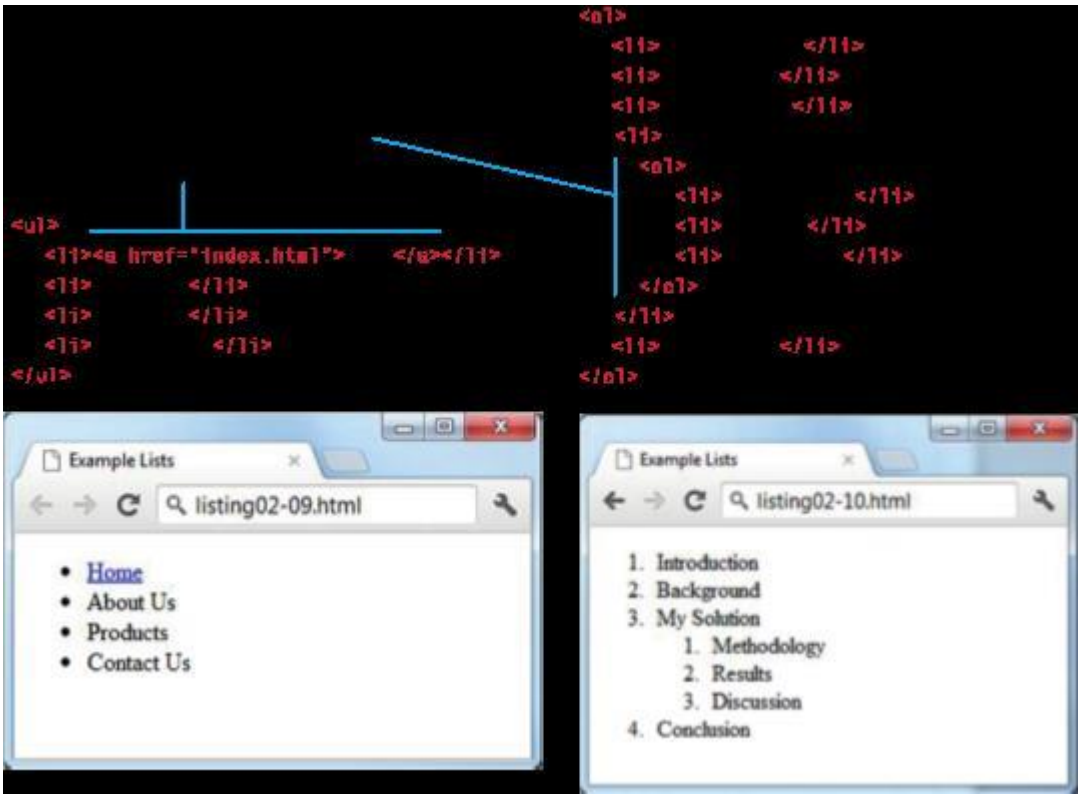


Figure: Nested lists (taken from Fundamentals of Web Development second edition 2017)

18.4 Description lists

Collection of name and description/definition pairs is called a description list. These lists tend to be used infrequently. Perhaps, the most common example would be an FAQ list. Unlike the other two lists (which contain <li> items within either a <ul> or <ol> parent container**), the container for a description**

list is the <dl> element. It contains <dt> (term or name to be described) and <dd> (describes each term) pairs for each item in the list.

```
<dl>
    <dt>HTML</dt>
    <dd>Hyper Text Markup Language</dd>
    <dt>CSS</dt>
    <dd>Cascading Style Sheet</dd>
</dl>
```

HTML

Hyper Text Markup Language

CSS

Cascading Style Sheet

Figure: Description lists (taken from Fundamentals of Web Development second edition 2017)

Module 19:

1. HTML Semantic Structuring

HTML semantic structuring adds meaning to a web page rather than merely making it look pretty. A <p> tag, for example, denotes that the contained text is a paragraph. Because people understand what paragraphs are and how to display them, this is both semantic and presentational. Tags like <b> and I on the other hand, do not have any semantic value. They merely specify how the text should be formatted (bold or italic) and add no further meaning to the markup.

Examples of semantic HTML tags include:

Header tags <h1> through <h6>

<blockquote>

<code>

There are many more semantic HTML tags to use as you build a standards-compliant website.

1.1. Advantages Semantic Structuring

Eliminating presentation-oriented markup and writing semantic HTML markup has a variety of important advantages:

1.1.1. ***Maintainability.***

Web pages with a lot of presentation markup are more difficult to edit and change than those with semantic markup. Our students are frequently startled to hear that maintaining and altering old code takes more time than writing new code. This is especially true for web initiatives. Web projects, in our experience, incur more "requirements drift" because of end user and customer input than traditional software development projects.

1.1.2. ***Performance.***

Semantic web pages are typically quicker to author and faster to download.

1.1.3. ***Accessibility.***

The content on web pages is not accessible to all web users. Voice reading software is used by users with vision problems to use the internet. If a website does not employ semantic markup, visiting it with speech reading software might be an extremely irritating experience. Many governments also require that websites for organizations that receive federal support follow particular accessibility criteria. The US government, for example, maintains its own set of Section 508 Accessibility Guidelines (http://www.section508.gov).

1.1.4. ***Search Engine Optimization***

The numerous search engine crawlers are, for many website owners, the most significant users of a website. Crawlers are automated programmes that explore the internet, looking for content to utilise in user searches. Semantic markup gives these crawlers better instructions: it tells them what material on the site is most important.

1.2. **Header Tag**

The header of a page or section is defined with the <header> tag. It usually includes a logo, a search engine, and navigational connections, among other things. This tag does not create a new outline section. The heading (an <h1>–<h6> element) of a surrounding section is typically contained in a <header> element. This, however, is not essential. One of the HTML5 elements is the header tag. Several <header> tags can be used in an HTML document, and they can be positioned in any area of it. It is forbidden to use the <header> tag inside the <footer> and <address> components, as well as inside another <header> tag.

The tag <header> is used in pairs. Between the opening (<header>) and closing (</header>) tags, the content is written.

1.3. Footer Tag

A web page's or section's footer is defined by the <footer> tag. It usually includes copyright information, contact information, navigation links, and so on. On a web page, there might be many <footer> tags. You can, for example, include a footer within the <article> element to keep information about the content (links, footnotes, etc.). Except for the <footer> and <header> tags, the tag can include other HTML components. One of the HTML5 components is the footer. If the footer has contact information, provide it in the <address> tag.

The <footer> element can contain the following:

- **copyright, authorship and contact information**
- **related documents**
- **sitemap**
- **back to top links**

The <footer> tag comes in pairs. The content is written between the opening (<footer>) and closing (</footer>) tags.

The <footer> tag supports the Global Attributes and the Event Attributes.

1.4. Navigation Tag

One of the HTML5 elements is the <nav> tag. It specifies a set of navigation links, either within the current content or to other papers. Tables of contents, menus, and indexes are examples of navigation blocks.

A single HTML document can include multiple <nav> tags, such as one for site navigation and another for intra-page navigation.

It's worth noting that the **<nav>** element doesn't contain all of the links in the HTML content; it only **contains the primary navigation blocks.** The nav> element can be used to define links in the website's footer, however the <footer> tag is more commonly used in such circumstances.

The <nav> tag is not allowed to be nested within the address> element.

The <nav> tag is a pair of tags. The material is written in the middle of the page.

1.5.Main Tag

In the HTML5 specification, the **<main> tag is a new block-level element**. The dominating content of the document's <body> is specified by this tag. The <main> tag's content must be unique and not duplicate blocks of the same type that appear in other pages, such as a site's header, header, footer, menu, search, copyright information, and so on. **The <article>, <aside>, <footer>, <header>, or <nav> tags cannot include the main> element. There can't be more than one <main> tag in a document that doesn't have a hidden attribute.**

Unlike elements like <body>, <h2>, and others, this element has no effect on the DOM's understanding of the page's structure. Landmarks can be used by assistive technology to easily identify and navigate to the document's big portions. It is preferable to utilise the <main> element instead of declaring role="main."

The "skipnav" technique allows users of assistive technology to skip huge chunks of repetitive material. This allows the user to quickly reach the primary material of a website. If you give the <main> element an id property, it will become the target of a skip navigation link.

The tag main> is used in pairs. Between the opening (<main>) and closing (</main>) tags, the text is written. The Global Attributes and Event Attributes are supported by the <main>tag.

1.6. Article Tag

One of the HTML5 elements is the <article> tag. It's used to describe stuff that's self-contained and self-contained. An article should have a distinct significance and stand out from the rest of the web page's material.

A blog entry, forum post, news item, or user remark can all be included in the <article> element.

The internal element represents an article that is associated to the external element when the <article> element is nested. The address> element can be used to provide author information for the <article> element. It will not, however, apply to nested <article> components. In a single HTML document, many article> tags can be used. The <h2> to <h6> tags can be nested within the <article> tag. If the <h1> tag has never been used previously, it can be used now. A <time> element with a datetime attribute can be used to express the <article> tag's publishing date and time.

The tag article> is used in pairs. Between the opening (article>) and closing (/article>) tags, the text is written.

1.7. Section Tag

One of the HTML5 components is **the <section>. It's utilised to make separate sections of a website with logically connected content** (news block, contact information, etc.). When designing a landing page, the <section> tag is frequently used to **break the page into logical blocks**.

A navigation menu, for example, must be wrapped in a <nav> tag, while a map display and a list of search results do not need to be wrapped in a <nav> tag and can be placed inside a <section>.

The nested <section> tag can be used to divide information into groups within the <article> tag. As a result, using <h1> -<h6> heads within the <article> and <section> tags is essential. The usage of a <h2> title in each section defined by the <section> tag is permitted. Instead of using the <section> element as a universal container for generating page structure, use the tag instead.

The tag <section> is used in pairs. Between the opening (<section>) and ending (</section>) tags, the text is written.

1.8. Figure Tag

The <figure> tag is used to indicate content that is self-contained. Diagrams, photos, illustrations, code samples, and so forth can all be included in the tag. One of the HTML5 elements is the figure tag. The <figure> tag's content is tied to the main flow's content. It is, nonetheless, regarded as a self-contained unit. The **<figcaption>** element is used to **add a caption or explanation** to the content of the figure> tag. The **<figcaption> tag is a child element of the figure> tag,** and it is either the first or last child element. The content will be shown at the top of the image if <figcaption> is the first nested element.

The tag <figure> is used in pairs. Between the beginning (<figure>) and closing (</figure>) tags, the content is written.

1.9. Caption Tag

The table's header is defined using the <caption> tag. The tag must be the first child of its parent <table> element and must be inside the <table> element immediately after the opening (<table>) tag. There is only one caption per table that can be defined. You must use the <figcaption> element instead of the <caption> element when the table> element containing the caption> is the only descendent of the <figure> element. A table caption is center-aligned above a table by default. However, the caption can be aligned and placed using the text-align and caption-side attributes.

The tag <caption> is used in pairs. Between the opening (<caption>) and closing (</caption>) tags, the text is written.

Module 20:

1. **Tools for web development**

HTML pages can be created in a variety of ways. An HTML editor can be used with any program that can edit and save text files. However, the use of the right technology can make writing web content much easier. You may have selected an HTML editor based on lab availability, familiarity, or some other factor.

While we will not be recommending specific tools for creating web content, we believe it is critical to understand the many types of web development tools and their relative benefits and drawbacks. **Web development tools are divided into five categories:**

- **WYSIWYG editors,**
- **Code editors,**
- **Full IDEs,**
- **Cloud-based environments,**
- **Code playgrounds.**

1. WYSIWYG editors.

Web tools that provide a user experience similar to that of a word processor are known as **What-You-See-Is-What-You-Get.** The benefit of such tools is that you don't require much (if any) HTML knowledge. However, there is a significant downside to using such technologies. These tools are never fully WYSIWYG, and they frequently have trouble previewing increasingly complex CSS. Indeed, in order to remedy such issues, these technologies nearly always have to give users with a typical HTML view. While we would never recommend relying only on such a tool, inexperienced end users may find them useful. In this genre, Adobe Dreamweaver and Adobe Muse are two prominent editors. WYSIWYG editors are also used in web-based publishing programs like blogs and content management systems.

1.2. Code editors.

Since web developers typically need knowledge of HTML, CSS, JavaScript, and more, many web developers prefer to use tools that allow them to focus on viewing and editing these text files. Nonetheless, it is helpful to use a tool that "understands" HTML, CSS, and so on. Such a tool might provide color coding, intelligent hints, tag completion, and so on. There are a wide range of choices in this genre, many of them open source. Some of the options include Atom, BlueFish, Brackets, Notepad++, Sublime Text, and Visual Studio Code.

1.3. Full IDEs.

Integrated Development Environments provide a fuller featured programming experience. They not only provide most of the same functionality as the previously mentioned code editors, but also typically provide **extra capabilities**, such as comprehensive help files, build tools, multiple-language support, and integration with other enterprise tools, such as databases. Some of the options in this genre include Eclipse, NetBeans, and Visual Studio. This extra power does come at a price, both figuratively and literally. The figurative costs is these complicated IDEs typically have a more substantial learning curve and can often have steep hardware requirements.

1.4. Cloud-based environments.

One of the fastest growing approaches to developing web applications is to do one's development, testing, and hosting all within an online environment. The key advantage of such an approach is that you don't have to worry about installing, supporting, and synchronizing different web development tools, since it is all done for you by the online environment. As well, using such online environments means that you don't really care what device you have; if you have an Internet connection, you can do your coding. Of course, that's also the key disadvantage. Since you need an Internet connection, you can't code while on the plane or in a forest (though these environments sometimes provide a mechanism for offline usage). At the time of writing, **CodeAnywhere and Cloud9 are two popular sites providing a complete IDE for web development.**

1.5. Code playgrounds.

Our final approach to web development tools also makes use of online environments. Code playgrounds are not about constructing complete sites. Instead, they **provide a way to experiment, demonstrate, and share smaller snippets of code**. Some of the most popular include CodePen , JSFiddle, and CSS Deck. These environments are especially valuable for students as a way to construct online portfolios and to show off their skills to prospective clients and employers.

Module 21:

Introduction to CSS:

In current web development best practices HTML should not describe the formatting or presentation of documents. Instead that presentation task is best performed using **Cascading Style Sheets (CSS).** CSS is a W3C standard for describing the appearance of HTML elements. Another common way to describe CSS's function is to say **that CSS is used to define the presentation of HTML documents.** With CSS, we can assign font properties, colors, sizes, borders, background images, and even position elements on the page. CSS can be added directly to any HTML element (via the style attribute), within the <head> element, or, most commonly, in a separate text file that contains only CSS.

Benefits of CSS

Before digging into the syntax of CSS, we should say a few words about why using CSS is a better way of describing appearances than HTML alone. The benefits of CSS include the following:

Improved control over formatting.

The degree of formatting control in CSS is significantly better than that provided in HTML. CSS gives web authors fine-grained control over the appearance of their web content.

Improved site maintainability.

Websites become significantly more maintainable because all formatting can be centralized into one CSS file, or a small handful of them. This allows you to make site-wide visual modifications by changing a single file.

Improved accessibility.

CSS-driven sites are more accessible. By keeping presentation out of the HTML, screen readers, and other accessibility tools work better, thereby providing a significantly enriched experience for those reliant on accessibility tools.

Improved page-download speed.

A site built using a centralized set of CSS files for all presentation will also be quicker to download because each individual HTML file will contain less style information and markup, and thus be smaller.

Improved output flexibility.

CSS can be used to adopt a page for different output media. This approach to CSS page design is often referred to as responsive design.

Module 22:

CSS Syntax

A CSS document consists of one or more style rules. **A rule consists of a selector that identifies the HTML element or elements that will be affected**, followed by a series of property:value pairs (each pair is also called a declaration).

Each individual CSS declaration must contain a property. These property names are predefined by the CSS standard. The **CSS2.1** recommendation defines over **a hundred different property names**, so some type of reference guide, whether in a book, online, or within your web development software, can be helpful.5 This section will only be able to cover most of the common CSS properties. The most used CSS properties are given below. Properties marked with an asterisk contain multiple subproperties not listed here (e.g., border-top, border-top color, border-top-width, etc).

Fonts

- font
- font-family
- font-size
- font-style
- font-weight
- @font-face

Text

- letter-spacing
- line-height
- text-align
- text-decoration*
- text-indent

Color and background

- background
- background-color
- background-image
- background-position
- background-repeat
- box-shadow
- color
- opacity

Borders

- border*
- border-color
- border-width
- border-style
- border-top, border-left, …*
- border-image*

- border-radius
- Spacing

padding

- padding-bottom, padding-left, …

margin

- margin-bottom, margin-left, …

Sizing

- height
- max-height
- max-width
- min-height
- min-width
- width

Layout

- bottom, left, right, top
- clear
- display
- float
- overflow
- position
- visibility
- z-index

Lists

- list-style*
- list-style-image
- list-style-type

Effects

- animation*
- filter
- perspective
- transform*
- transition*

Module 23:

CSS Syntax (Values)

Each CSS declaration also contains a value for a property. The unit of any given value is dependent upon the property. Some property values are from a predefined list of keywords. Others are values such as length measurements, percentages, numbers without units, color values, and URLs.

Color Values

Name

Use one of **17 standard color names**. **CSS3 has 140 standard names**.

color: red; color: hotpink; /* CSS3 only */

RGB

Uses three different numbers between 0 and 255 to describe the red, green, and blue values of the color.

color: rgb(255,0,0); color: rgb(255,105,180);

Hexadecimal

Uses a six-digit hexadecimal number to describe the red, green, and blue value of the color; each of the three **RGB values is between 0 and FF** (which is 255 in decimal). Notice that the hexadecimal number is preceded by **a hash or pound symbol (#).**

color: #FF0000; color: #FF69B4;

RGBa

This defines a partially transparent background color. **The "a" stands for "alpha,"** which is a term used to identify a transparency that is a **value between 0.0 (fully transparent) and 1.0 (fully opaque).**

color: rgba(255,0,0,0.5);

HSL

Allows you to specify a color using Hue Saturation and Light values. This is **available only in CSS3**. HSLA is also available as well.

color: hsl(0,100%,100%); color: hsla(330,59%,100%,0.5);

Just as there are multiple ways of specifying color in CSS, so too there are multiple ways of specifying a unit of measurement. these units can sometimes be complicated to work with. When working with print design, we generally make use of straightforward absolute units such as inches or centimeters and picas or points. However, because different devices have differing physical sizes as well as different pixel resolutions and because the user is able to change the browser size or its zoom mode, these absolute units don't always make sense with web element

measures.

Below is a list of the different units of measure in CSS. Some of these are relative units, in that they are based on the value of something else, such as the size of a parent element. Others are absolute units; in that they have a real-world size. Unless you are defining a style sheet for printing, it is recommended you avoid using absolute units. Pixels are perhaps the one popular exception (though, as we shall see later, there are also good reasons for avoiding the pixel unit). In general, most of the CSS that you will see uses either px, em, or % as a measure unit.

Units of Measure Values

px

Pixel. In CSS2 this is a relative measure, while in CSS3 it is absolute (1/96 of an inch).

em

Equal to the computed value of the font-size property of the element on which it is used. When used for font sizes, the em unit is in relation to the font size of the parent.

%

A measure that is always relative to another value. The precise meaning of % varies depending upon the property in which it is being used.

ex

A rarely used relative measure that **expresses size** in relation to the x-height of an element's font. Relative

ch

Another rarely used relative measure; this one express size in relation to the width of the zero (“0”) **character** of an element's font.

rem

Stands for **root em,** which is the font size of the root element. Unlike em, which may be different for each element, the rem is constant throughout the document.

vw, vh

Stands for **viewport width and viewport height**. Both are percentage values **(between 0 and 100)** of the viewport (browser window). This allows an item to change size when the viewport is resized.

in

Inches Absolute

cm

Centimeters Absolute

mm

Millimeters Absolute

pt

Points (equal to 1/72 of an inch)

Pc

Pica (equal to 1/6 of an inch)

Module 24:

Location of styles

As mentioned earlier, **CSS style rules can be in three different locations**. These three are not mutually exclusive, in that you could place your style rules in all three. In practice, however, web authors tend to place all their style definitions in one (or more) external style sheet files.

Inline Styles:

Inline styles are style rules placed within an HTML element via the style attribute. An inline style only affects the element it is defined within and overrides any other style definitions for properties used in the inline style. Notice that a selector is not necessary with inline styles and that semicolons are only required for separating multiple rules. Using inline styles is generally discouraged since they increase bandwidth and decrease maintainability (because presentation and content are intermixed and because it can be difficult to make consistent inline style changes across multiple files). Inline styles can, however, be handy for quickly testing out a style change.

Embedded Style Sheet:

Embedded style sheets (also called internal styles) are style rules placed within the <style> element (inside the <head> element of an HTML document). While better than inline styles, using embedded styles is also by and large discouraged. Since each HTML document has its own <style> element, it is more difficult to consistently style multiple documents when using embedded styles. Just as with inline styles, embedded styles can, however, be helpful when quickly testing out a style that is used in multiple places within a single HTML document. We sometimes use embedded styles in the book or in lab materials for that

reason.

External Style Sheet

External style sheets are style rules placed within a external text file with the .css extension. This is by far the most common place to locate style rules because it provides the best maintainability. When you make a change to an external style sheet, all HTML documents that reference that style sheet will automatically use the updated version. The browser can cache the external style sheet, which can improve the performance of the site as well.

Module 25:

Selectors

As teachers, we often need to be able to relay a message or instruction to either individual students or groups of students in our classrooms. In spoken language, we have a variety of different approaches we can use. We can identify those students by saying things like: “all of you talking in the last row,” or “all of you sitting in an aisle seat,” or “all of you whose name begins with ‘C’,” or “all first-year students,” or “John Smith.” Each of these statements identifies or selects a different (but possibly overlapping) set of students. Once we have used our student selector, we can then provide some type of message or instruction, such as “talk more quietly,” “hand in your exams,” or “stop texting while I am speaking.” when defining CSS rules, you will need to first use a selector to tell the browser which elements will be affected by the property values. CSS selectors allow you to select individual or multiple HTML elements. The topic of selectors has become more complicated than it was when we started teaching CSS in the late 1990s. There are now a variety of new selectors that are supported by most modern browsers. Before we get to those, let us look at the three basic selector types that have been around since the earliest CSS2 specification.

Element Selectors

Element selectors select all instances of a given HTML element. You can select all elements by using the universal element selector, which is the * (asterisk) character. You can select a group of elements by separating the different element names with commas. This is a sensible way to reduce the size and complexity of your CSS files, by combining multiple identical rules into a single rule.

Class Selectors

A class selector allows you to simultaneously target different HTML elements regardless of their position in the document tree. If a series of HTML elements have been labeled with the same class attribute value, then you can target them for styling by using a class selector, which takes the form: period (.) followed by the class name.

Id Selectors

An id selector allows you to target a specific element by its id attribute regardless of its type or position. If an HTML element has been labeled with an id attribute, then you can target it for styling by using an id selector, which takes the form: pound/hash (#) followed by the id name.

Module 26:

Attribute Selectors

An attribute selector provides a way to select HTML elements either by the presence of an element attribute or by the value of an attribute. This can be a very powerful technique, but because of uneven support by some of the browsers in the past, not all web authors have used them. Followings are the most common ways one can construct attribute selectors in CSS3.

[]

A specific attribute.

[title]

Matches any element with a title attribute

[=]

A specific attribute with a specific value. a[title="posts from this country"] Matches any <a> element whose title attribute is exactly "posts from this

country"

[~=]

A specific attribute whose value matches at least one of the words in a space delimited list of words. [title~="Countries"] Matches any title attribute that contains the word "Countries"

[^=]

A specific attribute whose value begins with a specified value. a[href^="mailto"] Matches any <a> element whose href attribute begins with "mailto"

[*=]

A specific attribute whose value contains a substring. img[src*="flag"] Matches any <img> element whose src attribute contains somewhere within it the text "flag"

[$=]

A specific attribute whose value ends with a specified value. a[href$=".pdf"] Matches any <a> element whose href attribute ends with the text ".pdf"

Module 27:

Pseudo-Element and Pseudo-Class Selectors

A pseudo-element selector is a way to select something that does not exist explicitly as an element in the HTML document tree but which is still a recognizable selectable object. For instance, you can select the first line or first letter of any HTML element using a pseudo-element selector. A pseudoclass selector does apply to an HTML element, but targets either a particular state or, in CSS3, a variety of family relationships. Below is a list of some of the

more common pseudo-class and pseudo-element selectors.

a:link pseudoclass

Selects links that have not been visited.

a:visited pseudoclass

Selects links that have been visited.

:focus pseudoclass

Selects elements (such as text boxes or list boxes) that have the input focus.

:hover pseudoclass

Selects elements that the mouse pointer is currently above.

:active pseudoclass

Selects an element that is being activated by the user. A typical example is a link that is being clicked.

:checked pseudoclass

Selects a form element that is currently checked. A typical example might be a radio button or a check box.

:firstchild pseudoclass

Selects an element that is the first child of its parent. A common use is to provide different styling to the first element in a list.

:firstletter pseudoelement

Selects the first letter of an element. Useful for adding drop-caps to a paragraph.

:firstline pseudoelement

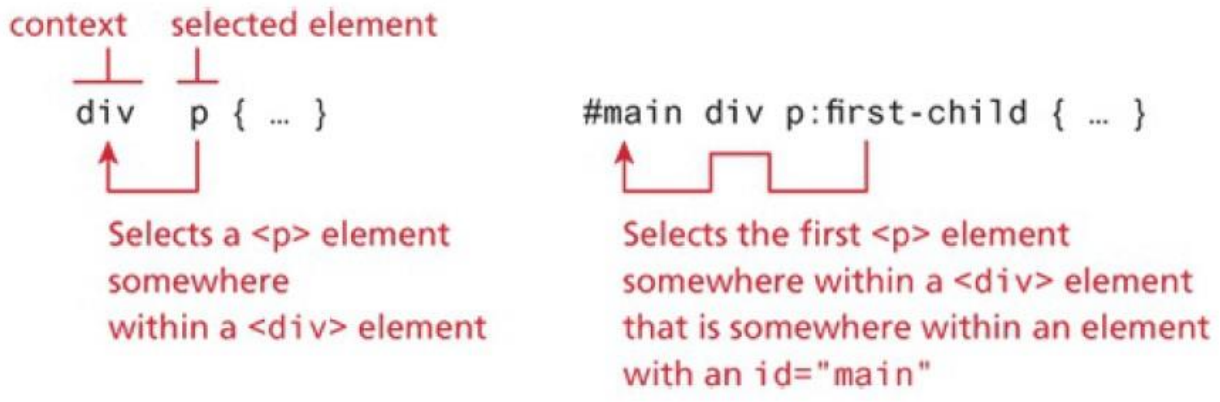
Selects the first line of an element.

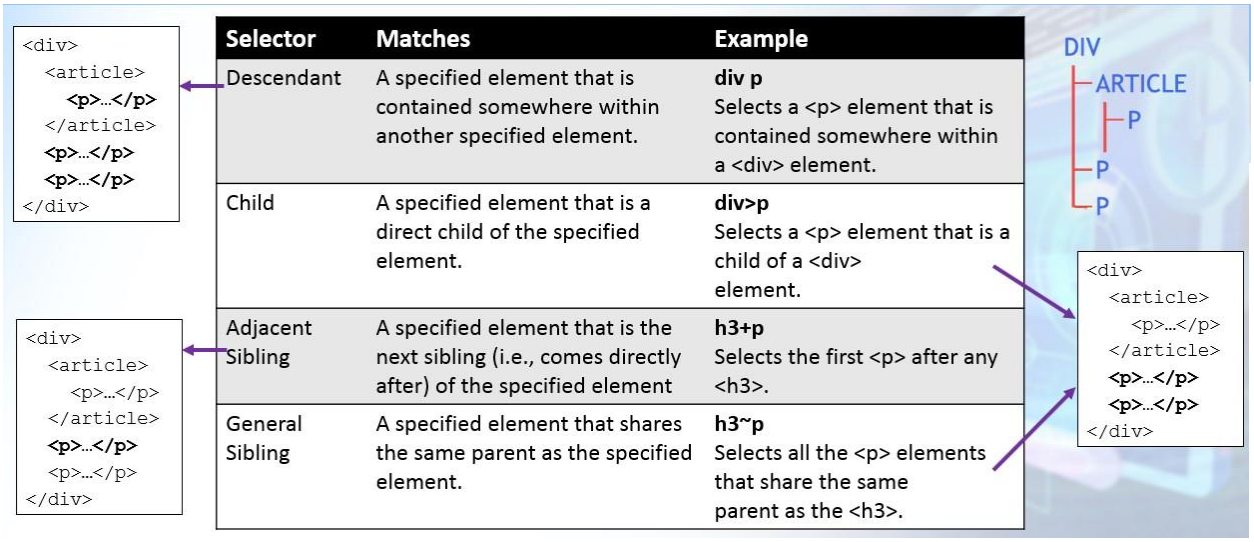
Module 28:

CSS Selectors (Contextual):

A contextual selector (in CSS3 also called combinators) allows you to select elements based on their *ancestors*, *descendants*, or *siblings*.

A descendant selector matches all elements that are contained within another element.





Module 29:

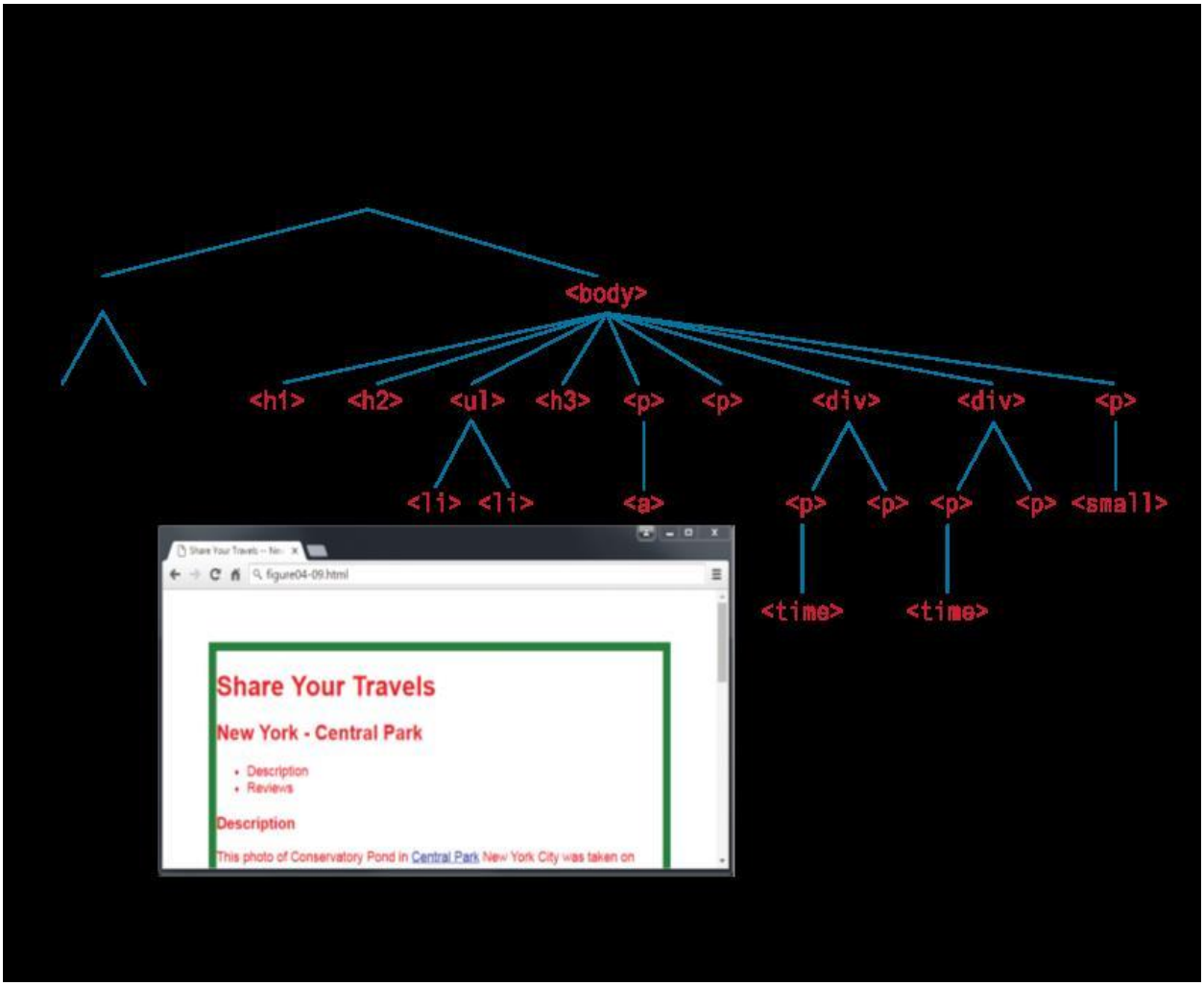
CSS Cascading:

- The “Cascade" in CSS refers to how conflicting rules are handled.
- The downward movement of water down a cascade is meant to be analogous to how a given style rule will continue to take precedence with child elements
- CSS uses the following cascade principles to help it deal with conflicts:
    - inheritance
    - specificity
    - location

Inheritance:

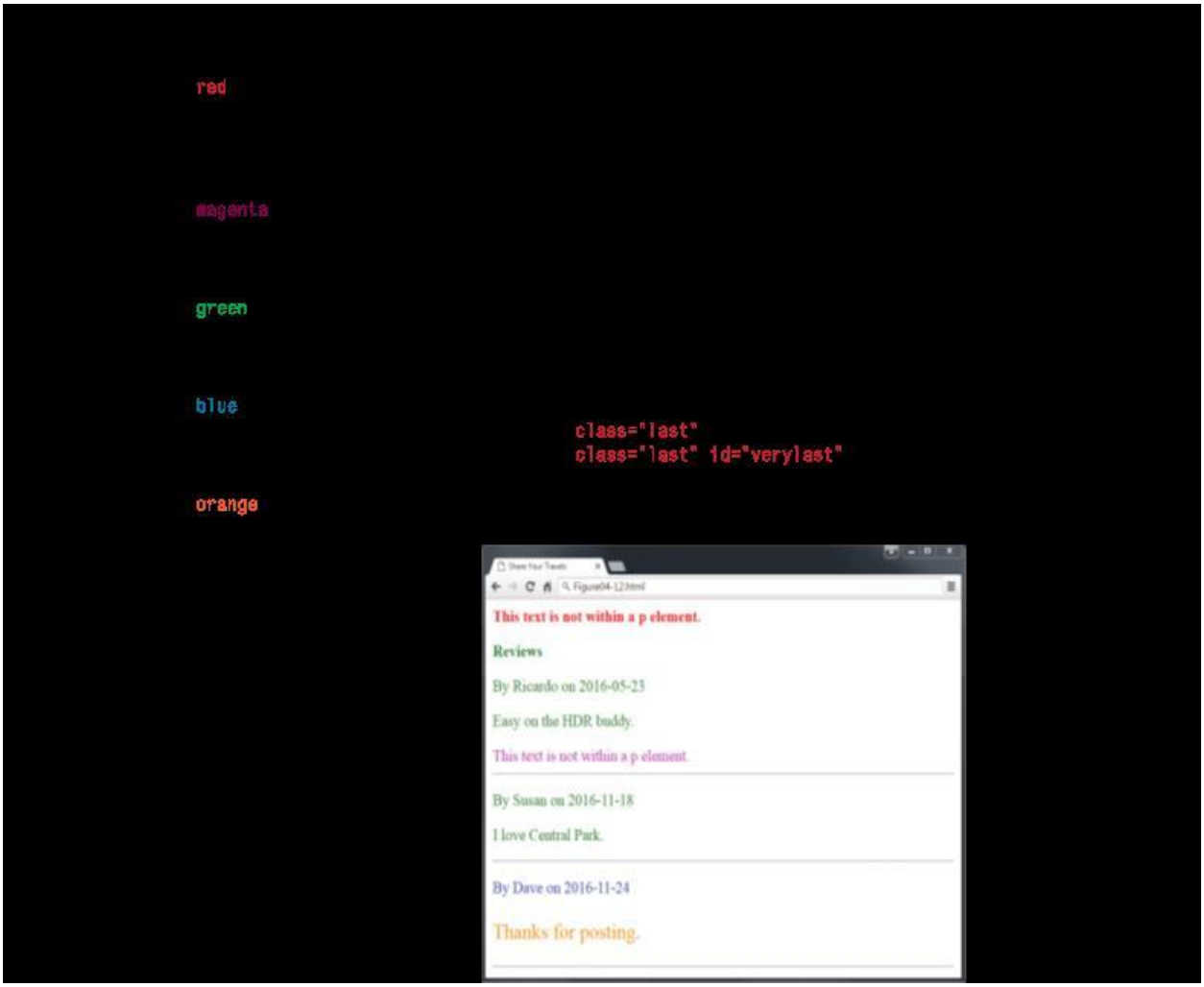
- Many (but not all) CSS properties affect not only themselves but their descendants as well
- Inheritable:
    - font, color, list, and text properties
- Not inheritable:
    - layout, sizing, border, background, and spacing properties

In the first example, only some of the property rules are inherited from the <body> element. That is, only the body element (thankfully!) will have a thick green border and the 100-px margin; however, all the text in the other elements in the document will be in the Arial font and colored red.



Specificity:

Specificity is how the browser determines which style rule takes precedence when more than one style rule could be applied to the same element. In CSS, the more specific the selector, the more it takes precedence (i.e., overrides the previous definition).



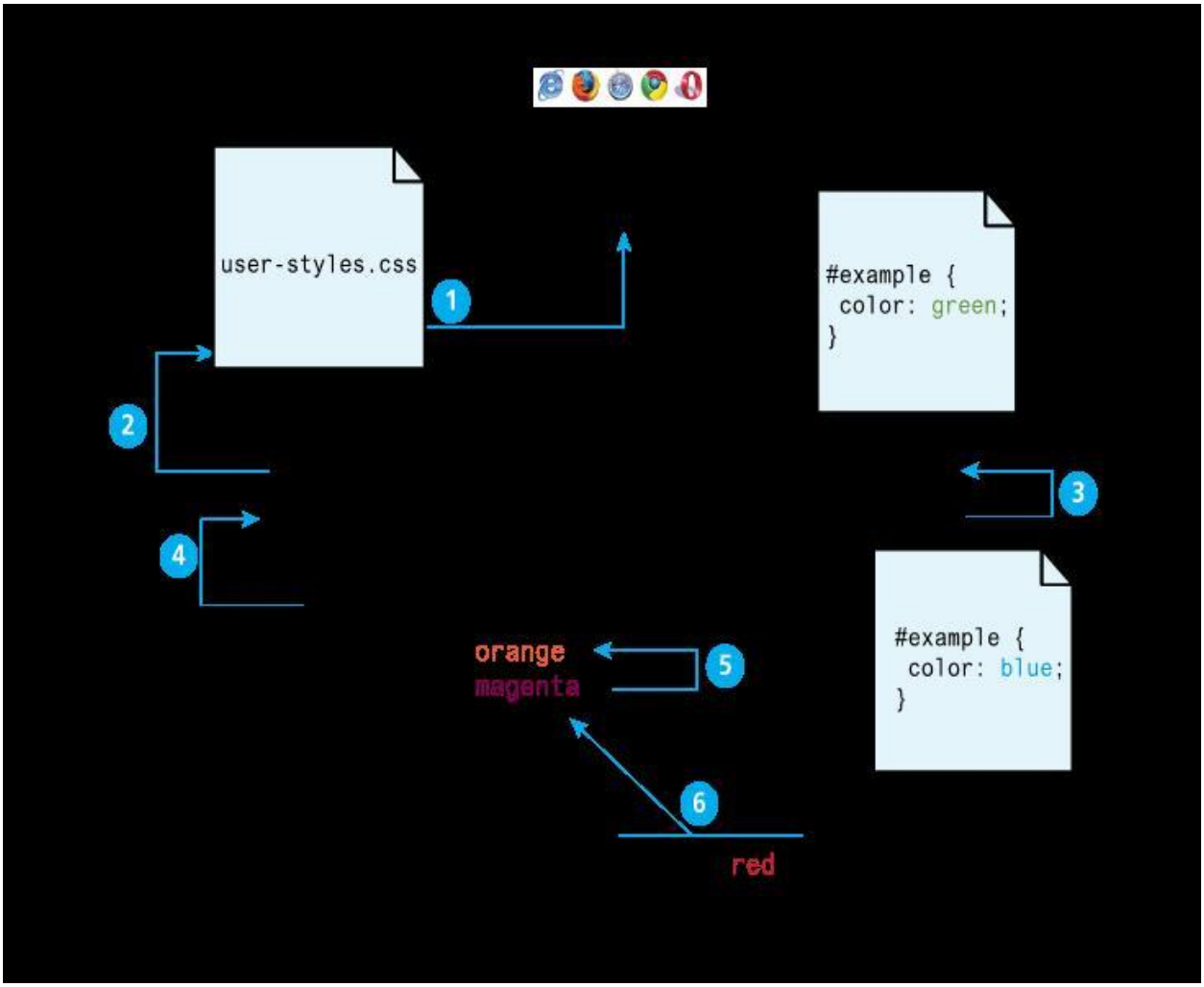
In the example shown in image, the color and font-weight properties defined in the <body> element are inheritable and thus potentially applicable to all the child elements contained within it. However, because the and <p> elements also have the same properties set, they *override* the value defined for the <body> element because their selectors (and <p>) are more specific. As a consequence, their font-weight is normal and their text is

colored either green or magenta.

Location:

When inheritance and specificity cannot determine style precedence, the principle of location will be used. The principle of location is that when rules have the same specificity, then the latest are given more weight. For instance, an inline style will override one defined in an external author style sheet or an embedded style sheet. Similarly, an embedded style will override

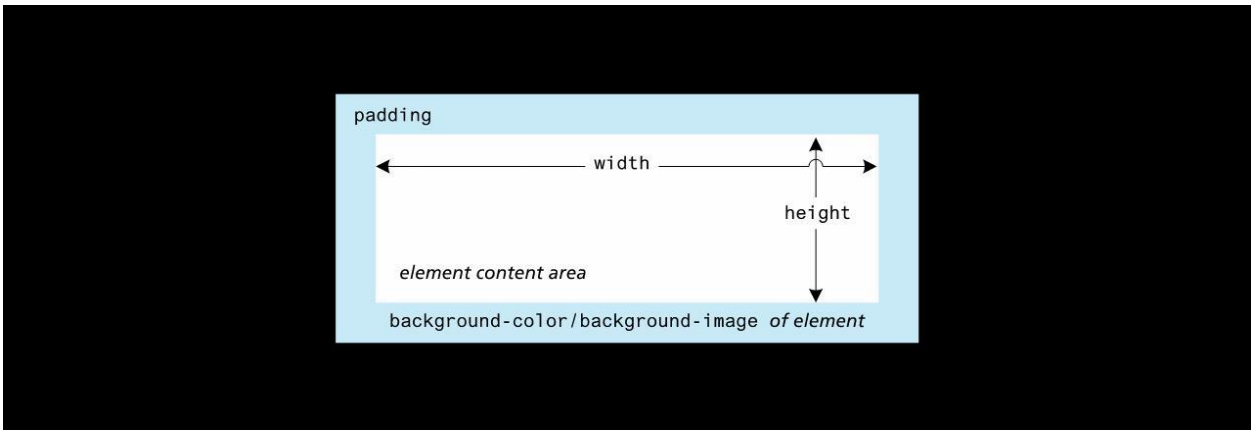
an equally specific rule defined in an external author style sheet if it appears after the external sheet's <link> element.

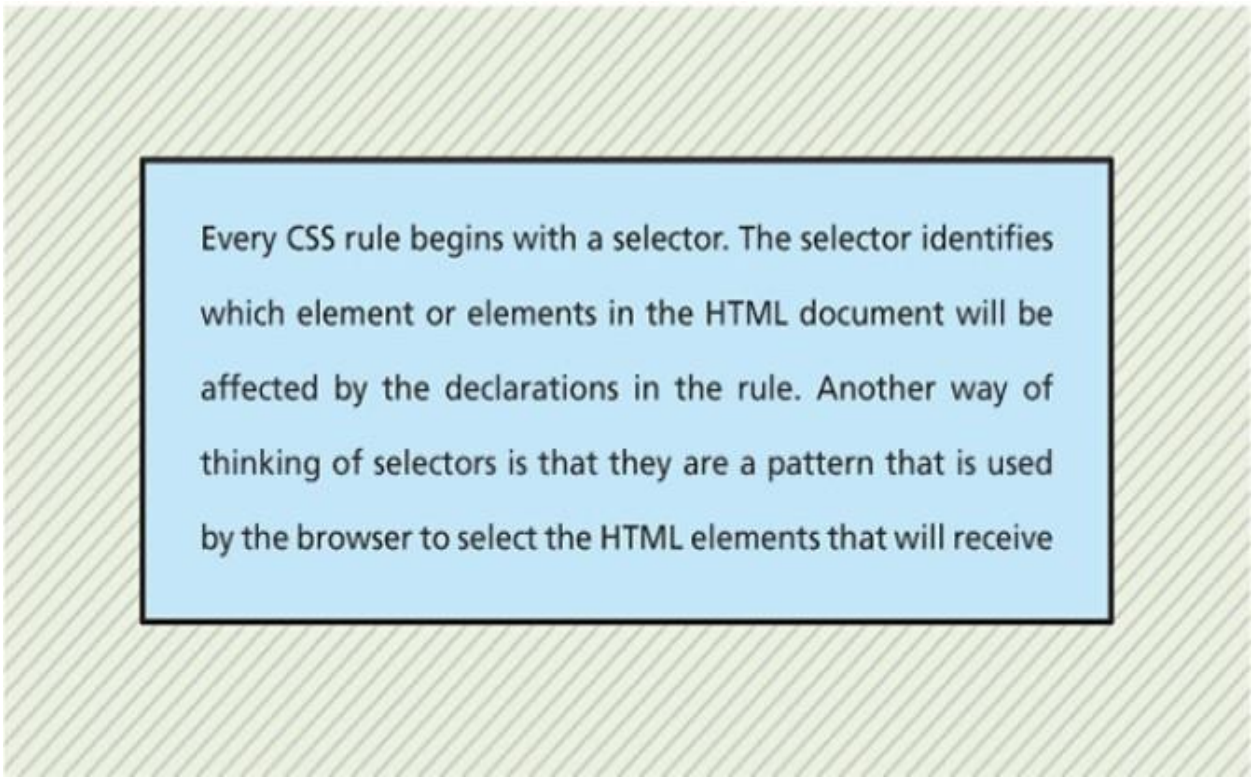

user-styles.css
1
#example {
color: green;
}
2
3
4
#example {
color: blue;
}
orange
magenta
5
6
red

Module 30:

The Box Model

In CSS, all HTML elements exist within an element box shown in Figure below. In order to become proficient with CSS, you must become familiar with the element box.





Background:

the background color or image of an element fills an element out to its border (if it has one, that is).

Common Background Properties

Reference: taken from https://www.w3schools.com/

Value	Description
background-color	Specifies the background color to be used
background-image	Specifies ONE or MORE background images to be used
background-position	Specifies the position of the background images
background-size	Specifies the size of the background images
background-repeat	Specifies how to repeat the background images
background-origin	Specifies the positioning area of the background images
background-clip	Specifies the painting area of the background images
background-attachment	Specifies whether the background images are fixed or scrolls with the rest of the page

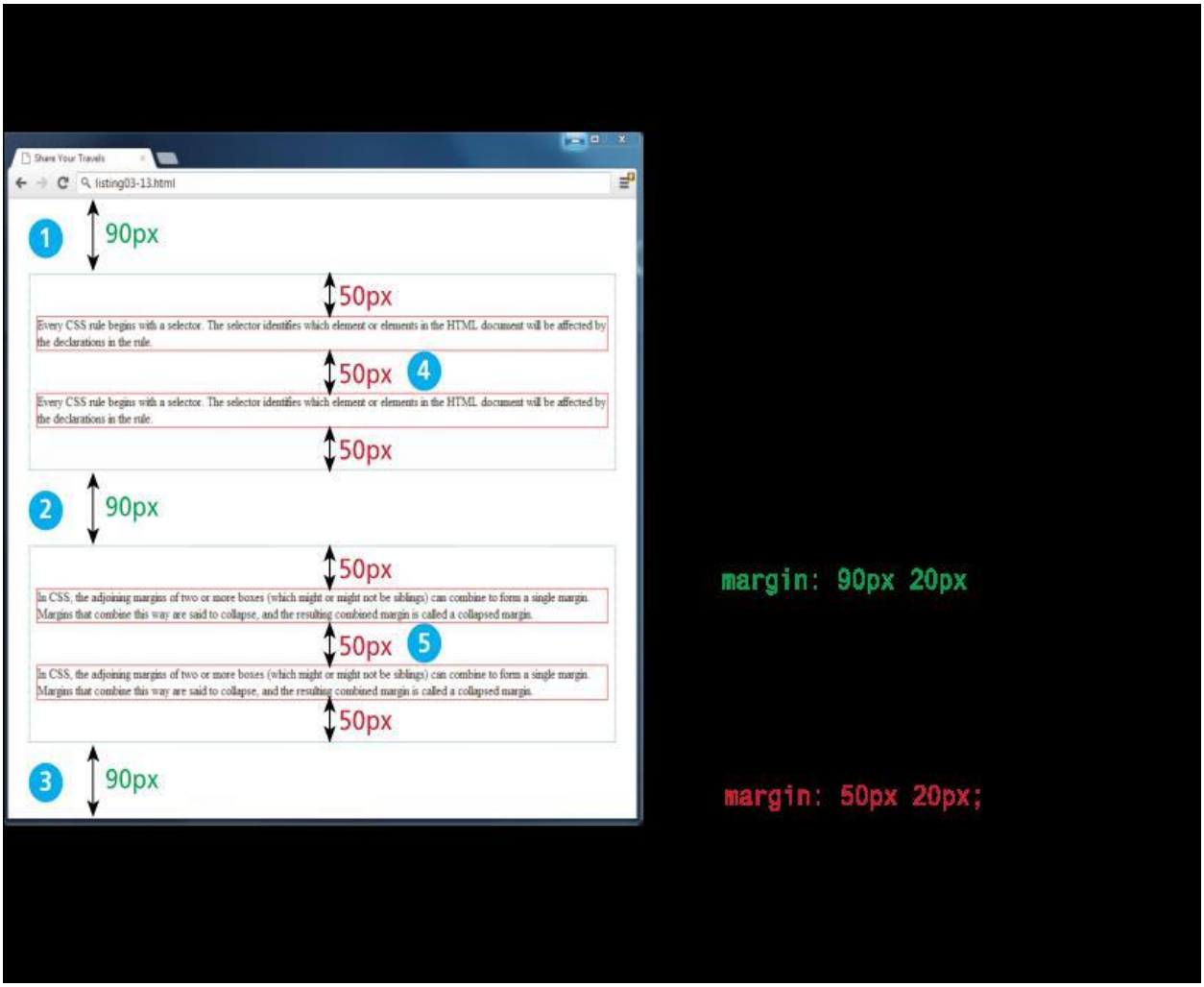
CSS TRBL Shortcut

- border-top-color: red; /* sets just the top side */
- border-right-color: green; /* sets just the right side */
- border-bottom-color: yellow; /* sets just the bottom side */
- border-left-color: blue; /* sets just the left side */

- Alternately, we can set all four sides at once:
- border-color: red; /* sets all four sides to red */
- border-color: red green orange blue; /* sets 4 colors */



Collapsing Margins



Module 31:

Box Dimensions

Content-box

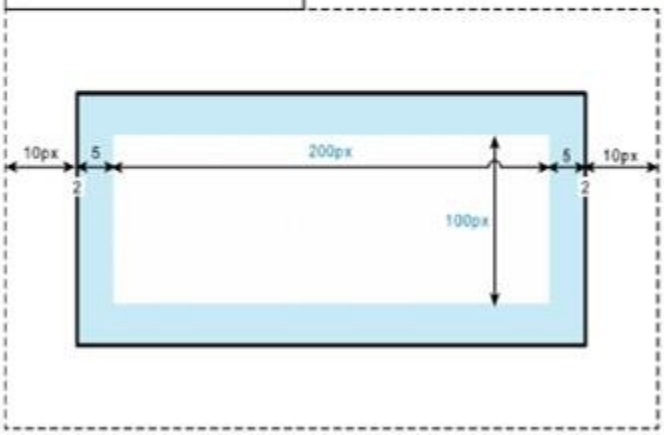
- width/height applied to contents only
- borders and padding counted separately
- margins counted separately

Border-box

- width/height include border and padding
- margins counted separately

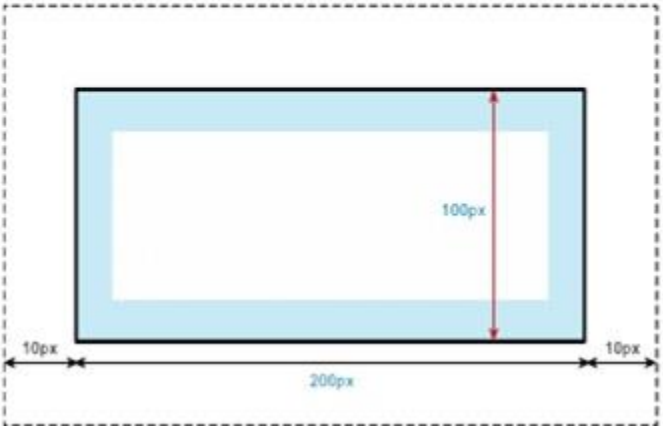
```
div {
  box-sizing: content-box;
  width: 200px;
  height: 100px;
  padding: 5px;
  margin: 10px;
  border: solid 2pt black;
}
```

True element width = 10 + 2 + 5 + 200 + 5 + 2 + 10 = 234 px
True element height = 10 + 2 + 5 + 100 + 5 + 2 + 10 = 134 px



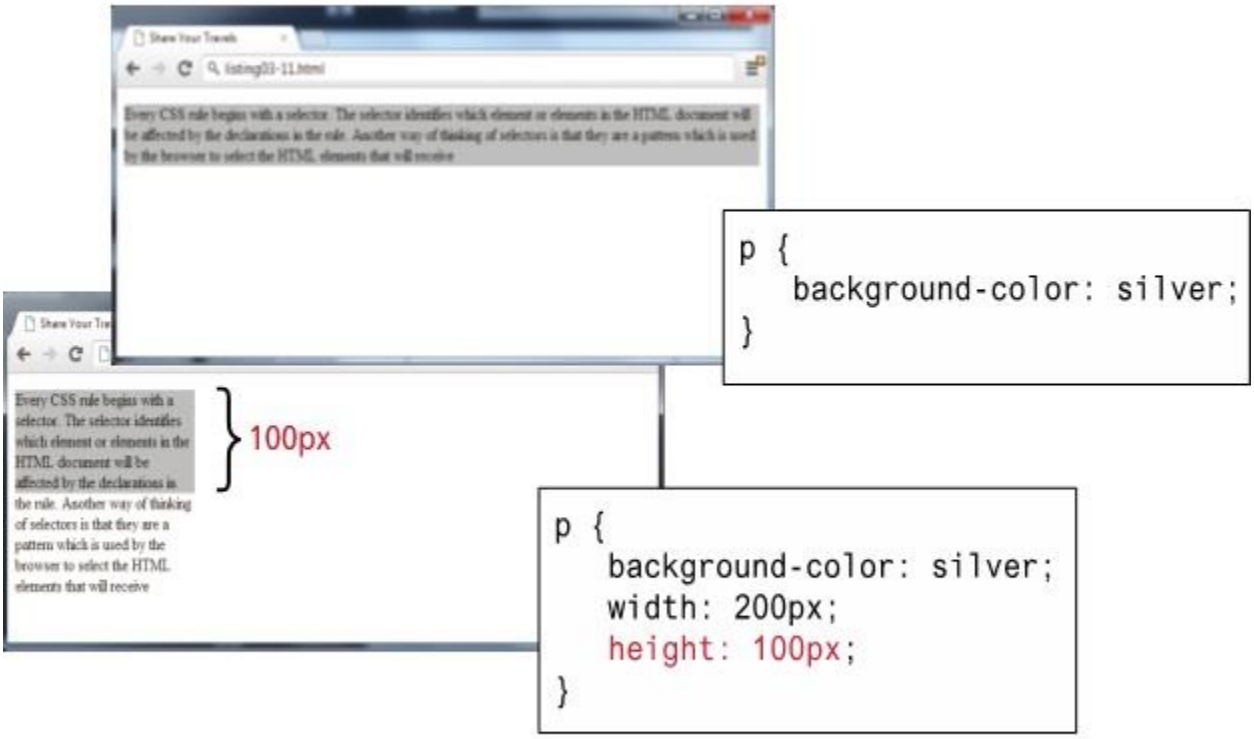

```
div {
  ...
  box-sizing: border-box;
}
```

True element width = 10 + 200 + 10 = 220 px
True element height = 10 + 100 + 10 = 120 px



Limitations of Height Property

For block-level elements such as <p> and elements, there are limits to what the width and height properties can actually do. You can shrink the width, but the content still needs to be displayed, so the browser may very well ignore the height that you set.

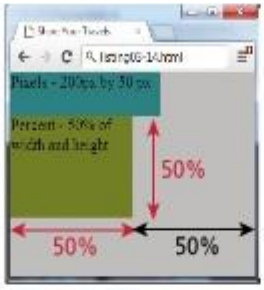


Overflow Property

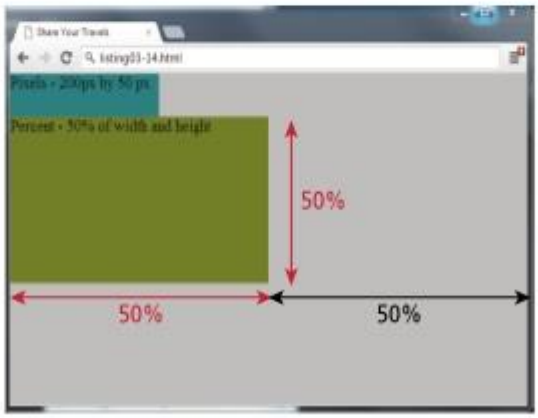
It is possible to control what happens with the content if the box's width and height are not large enough to display the content using the overflow property.

Box Sizing Using Percentages

```
<style>
  html,body {
     margin:0;
     width:100%;
     height:100%;
     background: silver;
  }
  .pixels {
     width:200px;
     height:50px;
     background: teal;
  }
  .percent {
     width:50%;
     height:50%;
     background: olive;
  }
```

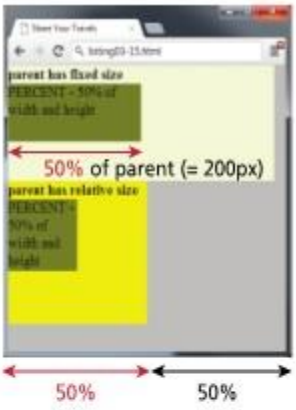



```
<body>
   <div class="pixels">
     Pixels - 200px by 50 px
   </div>
   <div class="percent">
     Percent - 50% of width and height
   </div>
</body>
```

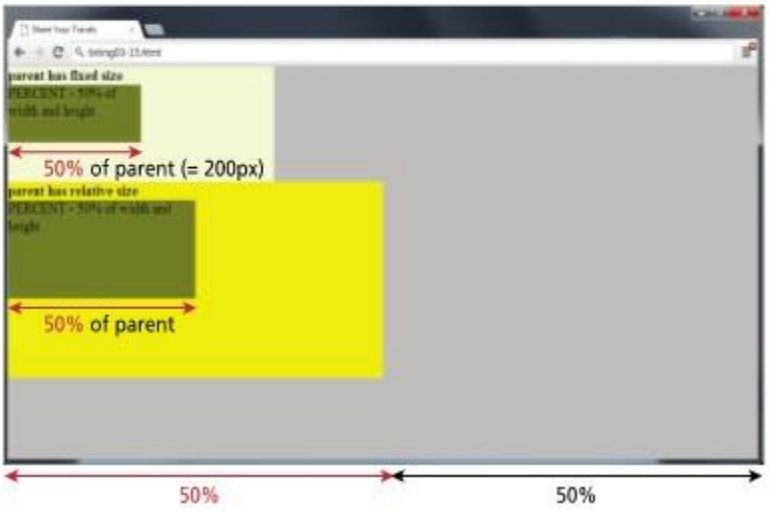


Box Sizing Using Pixels

```
   .parentFixed {
      width:400px;
      height:150px;
      background: beige;
   }
   .parentRelative {
      width:50%;
      height:50%;
      background: yellow;
   }
</style>
```




```
<body>
<div class="parentFixed">
   <strong>parent has fixed size</strong>
   <div class="percent">
      PERCENT - 50% of width and height
   </div>
</div>
<div class="parentRelative">
   <strong>parent has relative size</strong>
   <div class="percent">
      PERCENT - 50% of width and height
   </div>
</div>
</body>
```



Module 32:

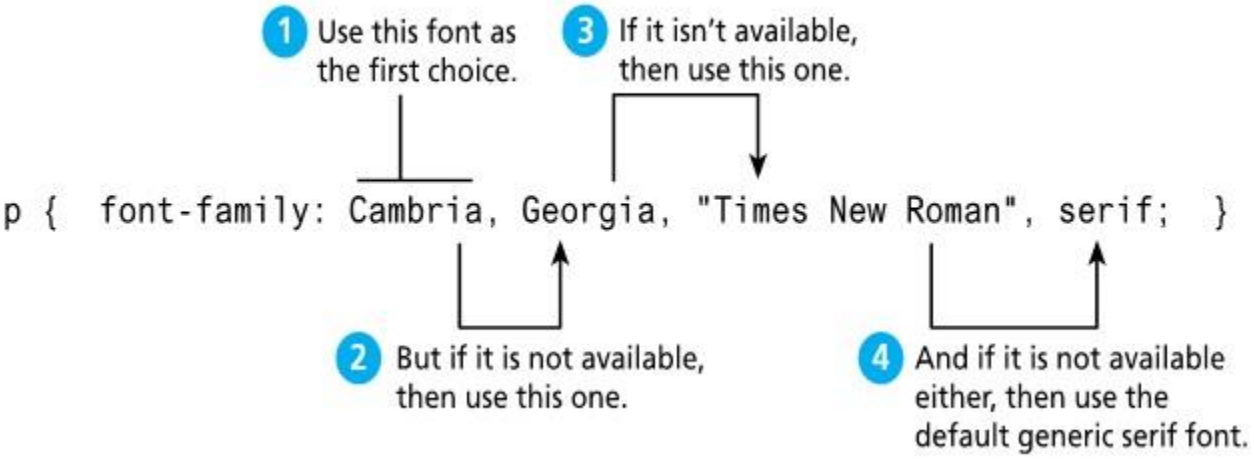
CSS Text Styling

CSS provides two types of properties that affect text. The first we call font properties because they affect the font and its appearance. The second type of CSS text properties are referred to here as paragraph properties since they affect the text in a similar way no matter which font is being used.

Font Properties

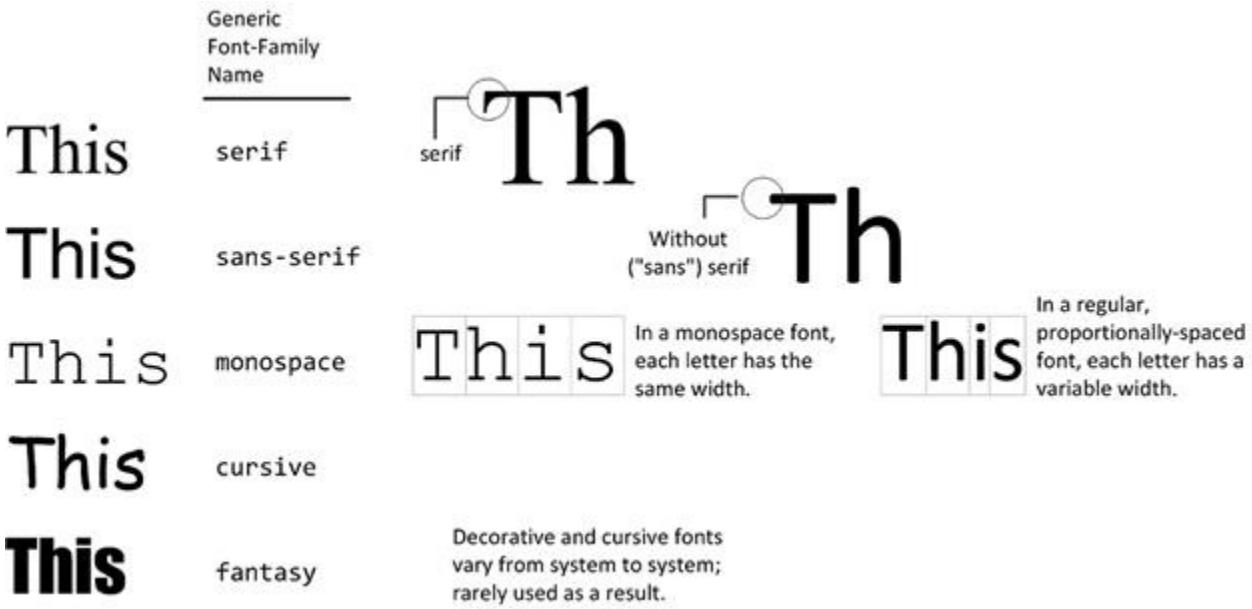
Property	Description
font	A combined shorthand property that allows you to set the family, style, size, variant, and weight in one property. **style weight variant size font-family**
font-family	Specifies the typeface/font to use. More than one can be specified.
font-size	The size of the font in one of the measurement units
font-style	Specifies whether italic, oblique, or normal
font-variant	Specifies either small-caps text or none
font-weight	Specifies either normal, bold, bolder, lighter, or a value between 100 and 900 in multiples of 100, where larger number represents weightier (i.e., bolder) text.

Specifying the Font Family



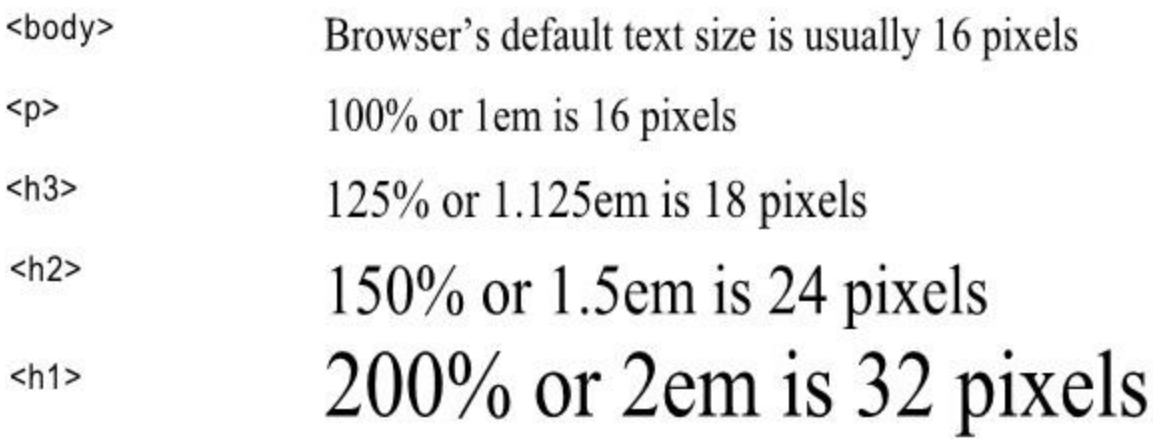
Different Font Families

The font-family property supports five different generic families; the browser supports a typeface from each family.



Font Sizes

Another potential problem with web fonts is font sizes. In a print-based program such as a word processor, specifying a font size is unproblematic. Making some text 12 pt will mean that the font's bounding box (which in turn is roughly the size of its characters) will be 1/6 of an inch tall when printed, while making it 72 pt will make it roughly one inch tall when printed. However, as we saw absolute units such as points and inches do not translate very well to pixel-based devices. Newer mobile devices in recent years have been increasing pixel densities so that a given CSS pixel does not correlate to a single device pixel.




```css
/* using 16px scale */

body { font-size: 100%; }
p { font-size: 1em; }        /* 1.0  x 16 = 16 */
h3 { font-size: 1.125em; }   /* 1.25 x 16 = 18 */
h2 { font-size: 1.5em; }     /* 1.5 x 16  = 24 */
h1 { font-size: 2em; }       /* 2 x 16 = 32 */
```

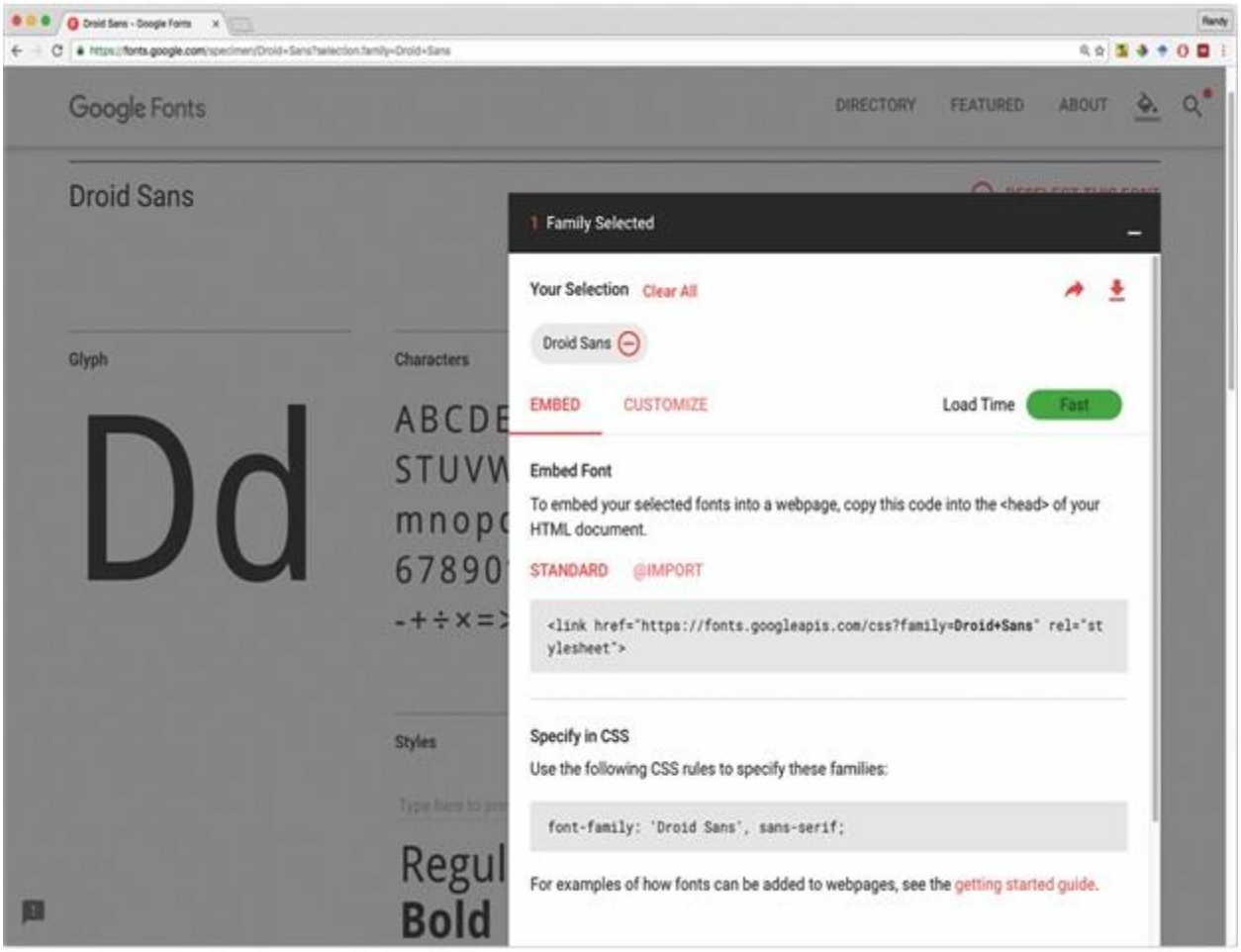
```html
<body>
  Browser's default text size is usually 16 pixels
  <p>100% or 1em is 16 pixels</p>
  <h3>125% or 1.125em is 18 pixels</h3>
  <h2>150% or 1.5em is 24 pixels</h2>
  <h1>200% or 2em is 32 pixels</h1>
</body>
```

CSS Text Styling

Embedding Fonts

Due to the ongoing popularity of open source font sites such as Google Web Fonts (https://fonts.google.com) and Font Squirrel (http://www.fontsquirrel.com/), @font-face seems to have gained a critical mass of widespread usage.

The following example illustrates how to use Droid Sans (a system font also used by Android devices) from Google Web Fonts using @font-face.



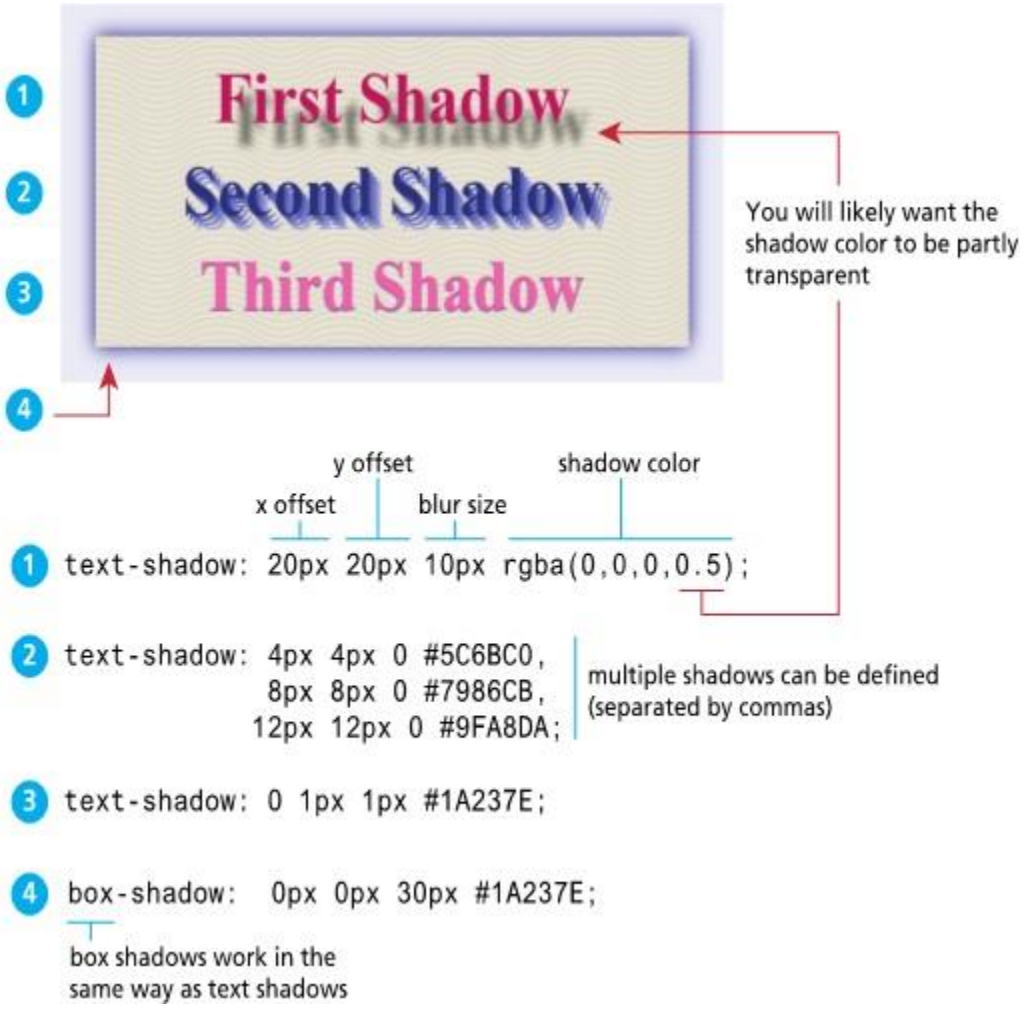
Paragraph Properties

Reference: taken from https://www.w3schools.com/

Property	**Description**
letter-spacing	Specifies the space between characters in a text
line-height	Specifies the line height
text-indent	Specifies the indentation of the first line in a text-block
white-space	Specifies how to handle white-space inside an element
word-spacing	Specifies the space between words in a text

text-shadow	Specifies the shadow effect added to text

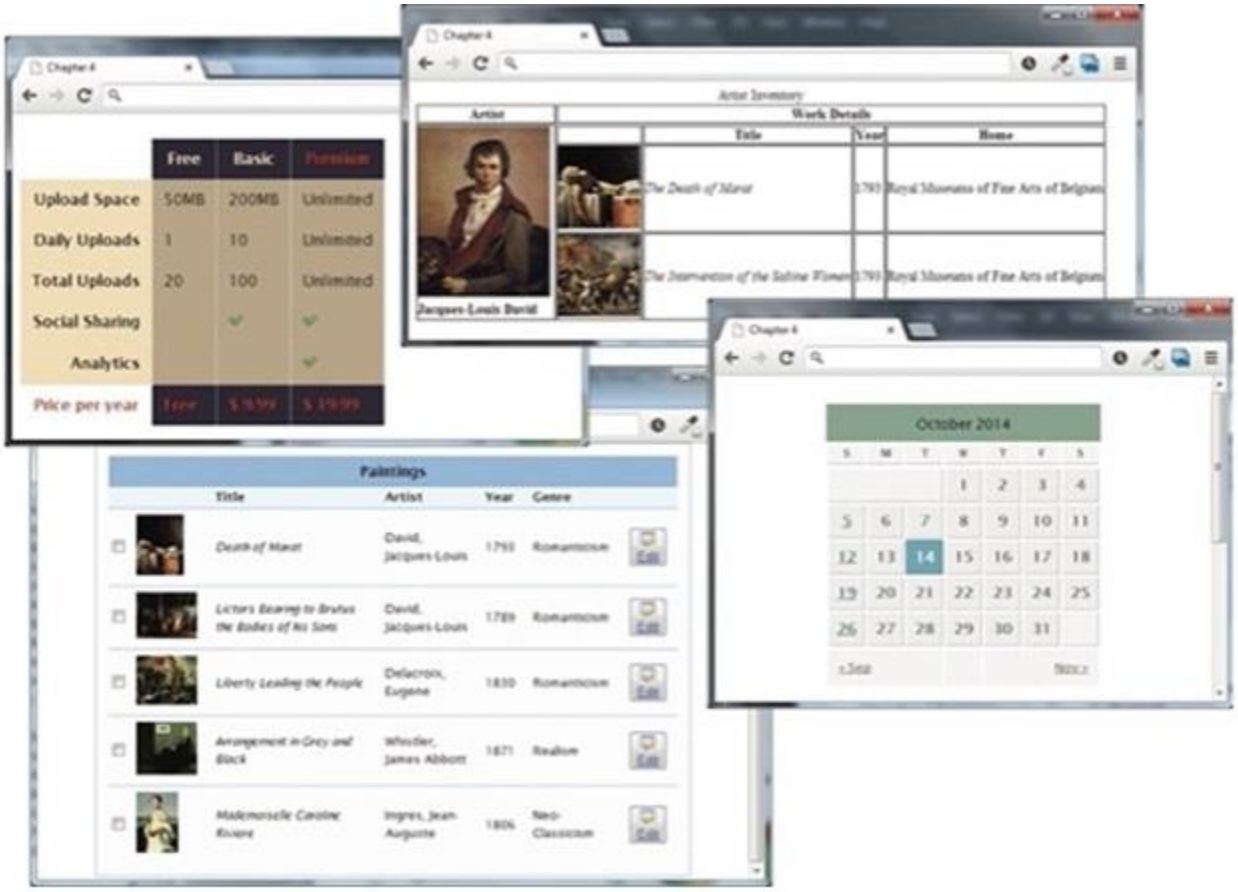
Shadow Example:



Module 33:

Introducing Tables

A table in HTML is created using the <table> element and can be used to represent information that exists in a two-dimensional grid. Tables can be used to display calendars, financial data, pricing tables, and many other types of data. Just like a real-world table, an HTML table can contain any type of data: not just numbers, but text, images, forms, even other tables, as shown in the Figure below.



Basic Table Structure

Create a Basic Table Complex Content in Tables As can be seen in Figure, an HTML <table> contains any number of rows (<tr>); each row contains any number of table data cells (<td>). The indenting shown in Figure is purely a convention to make the markup more readable by humans.

<table>
<tr>
<td>The Death of Marat</td>
<td>Jacques-Louis David</td>
<td>1793</td>
<td>162cm</td>
<td>128cm</td>
</tr>
<tr>
<td>Burial at Ornans</td>
<td>Gustave Courbet</td>
<td>1849</td>
<td>314cm</td>
<td>663cm</td>
</tr>
</table>

```
<table>
   <tr>
      <td>The Death of Marat</td>
      <td>Jacques-Louis David</td>
      <td>1793</td>
      <td>162cm</td>
      <td>128cm</td>
   </tr>
   <tr>
      <td>Burial at Ornans</td>
      <td>Gustave Courbet</td>
      <td>1849</td>
      <td>314cm</td>
      <td>663cm</td>
   </tr>
</table>
```



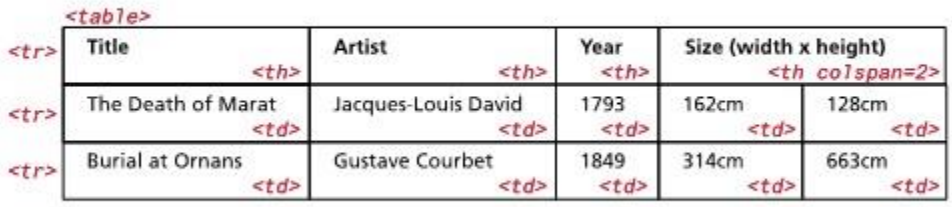
Adding Headings

Many tables will contain some type of headings in the first row. In HTML, you indicate header data by using the <th> instead of the <td> element.

Spanning Rows and Columns

If you want a given cell to cover several columns or rows, then you can do so by using the colspan or rowspan attributes.

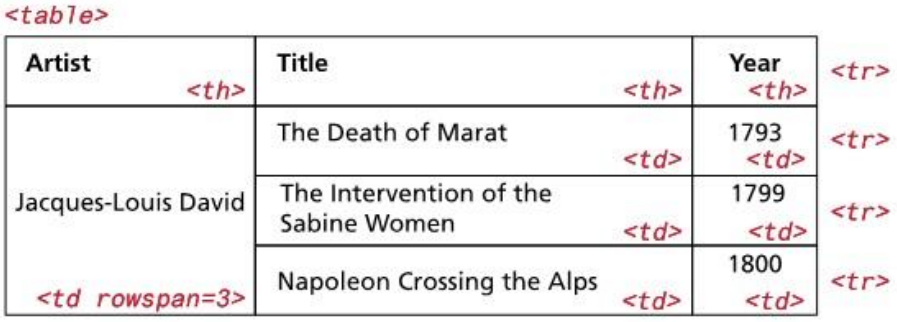
Spanning columns



Notice that this row now only has four cell elements.

```
<table>
   <tr>
      <th>Title</th>
      <th>Artist</th>
      <th>Year</th>
      <th colspan="2">Size (width x height)</th>
   </tr>
   <tr>
      <td>The Death of Marat</td>
      <td>Jacques-Louis David</td>
      <td>1793</td>
      <td>162cm</td>
      <td>128cm</td>
   </tr>
   ...
</table>
```

Spanning Rows



Notice that these two rows now only have two cell elements.

```
<table>
   <tr>
      <th>Title</th>
      <th>Artist</th>
      <th>Year</th>
   </tr>
   <tr>
      <td rowspan="3">Jacques-Louis David</td>
      <td>The Death of Marat</td>
      <td>1793</td>
   </tr>
   <tr>
      <td>The Intervention of the Sabine Women</td>
      <td>1799</td>
   </tr>
   <tr>
      <td>Napoleon Crossing the Alps</td>
      <td>1800</td>
   </tr>
</table>
```

Module 34:

Introducing Tables

Additional Table Elements

While the previous sections cover the basic elements and attributes for most simple tables, there are some additional table elements that can add additional meaning and accessibility to one's tables.

A title for the table is good for accessibility.

These describe our columns, and can be used to aid in styling.

Table header could potentially also include other <tr> elements.

Yes, the table footer comes *before* the body.

Potentially, with styling the browser can scroll this information, while keeping the header and footer fixed in place.

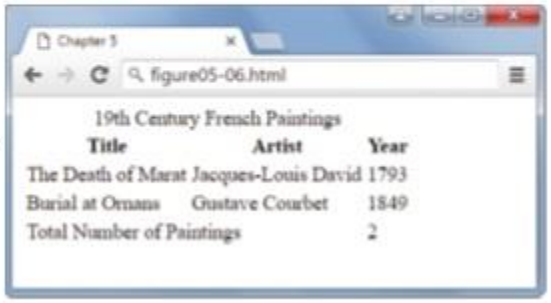
```
<table>
    <caption>19th Century French Paintings</caption>
    <col class="artistName" />
    <colgroup id="paintingColumns">
        <col />
        <col />
    </colgroup>

    <thead>
        <tr>
            <th>Title</th>
            <th>Artist</th>
            <th>Year</th>
        </tr>
    </thead>

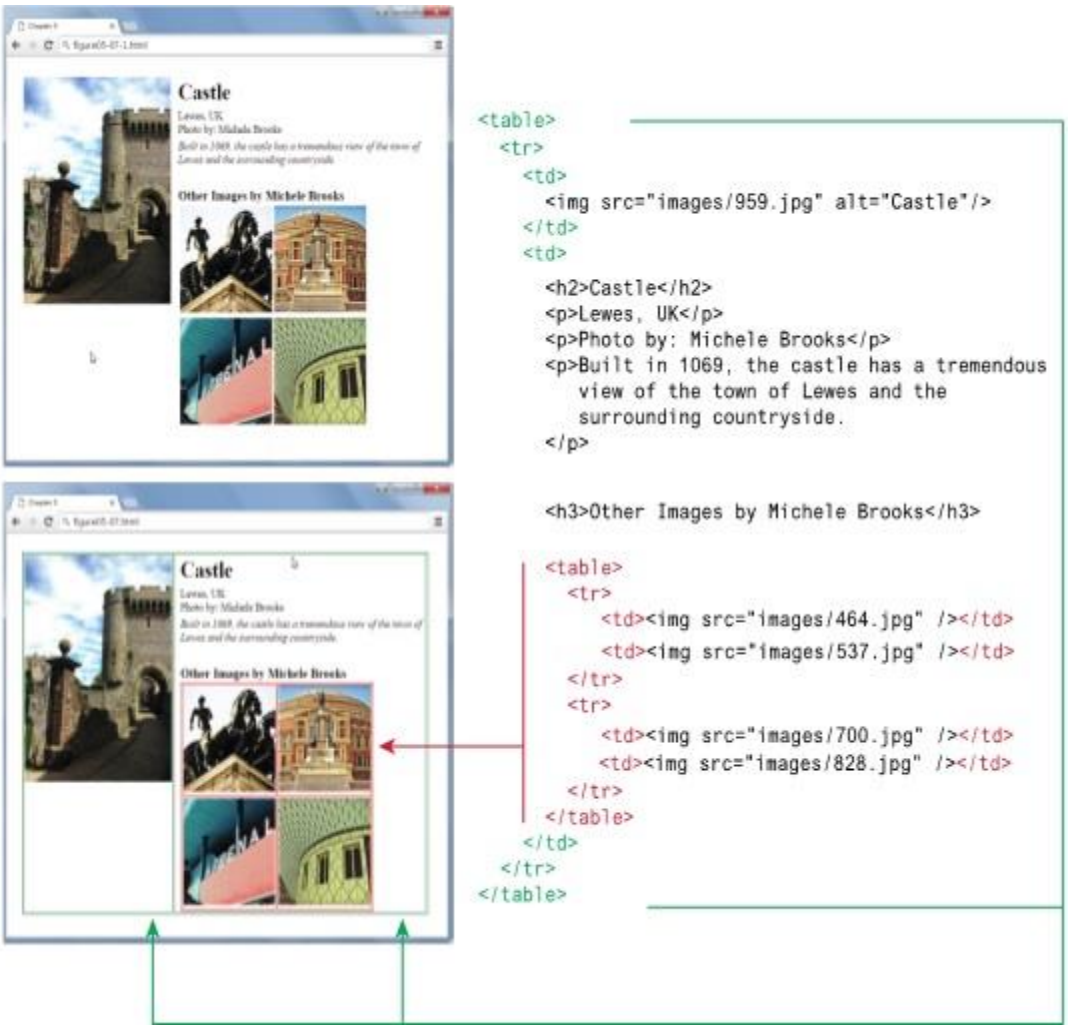
    <tfoot>
        <tr>
            <td colspan="2">Total Number of Paintings</td>
            <td>2</td>
        </tr>
    </tfoot>

    <tbody>
        <tr>
            <td>The Death of Marat</td>
            <td>Jacques-Louis David</td>
            <td>1793</td>
        </tr>
        <tr>
            <td>Burial at Ornans</td>
            <td>Gustave Courbet</td>
            <td>1849</td>
        </tr>
    </tbody>

</table>
```



Using Tables for Layout



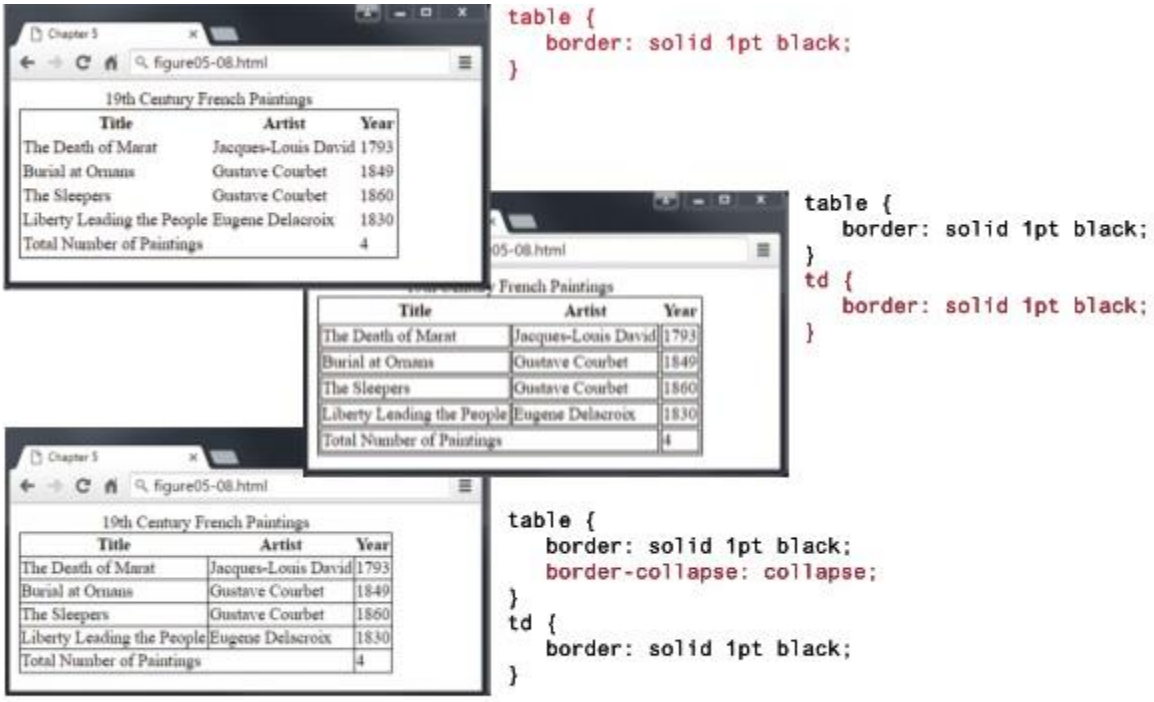
Module 35:

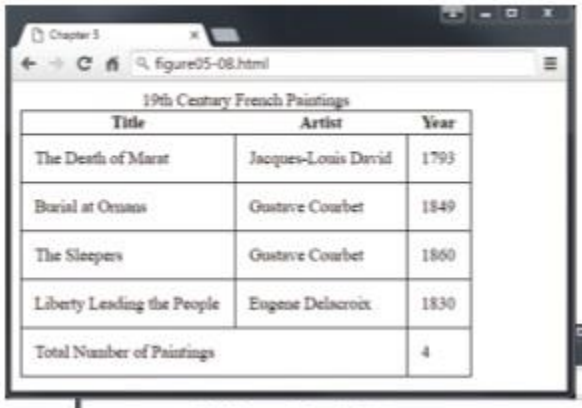
Styling Tables

You can style the tables in different way outs.

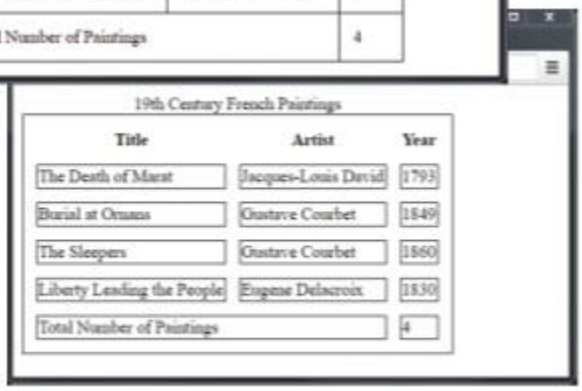
Table Borders

Following are some examples of the table with different borders:

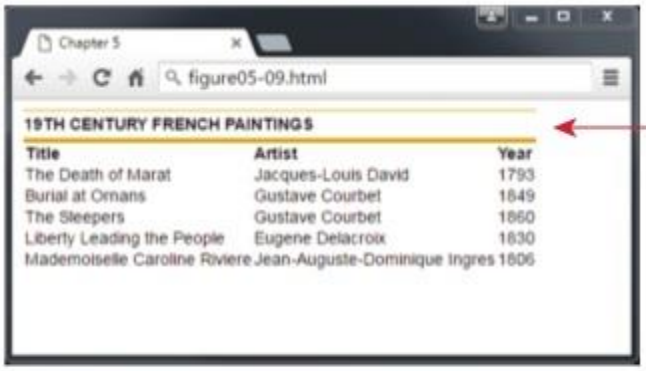




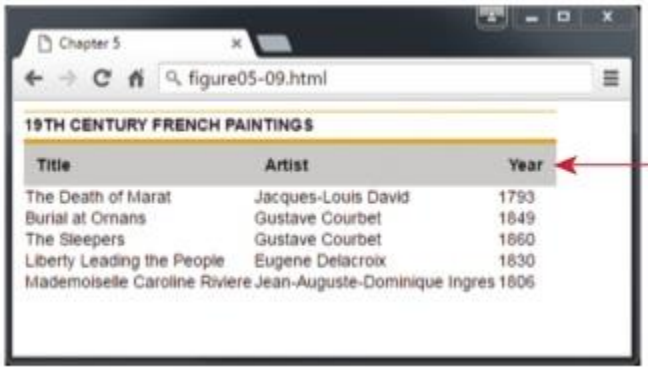

```
table {
   border: solid 1pt black;
   border-collapse: collapse;
}
td {
   border: solid 1pt black;
   padding: 10pt;
}
```



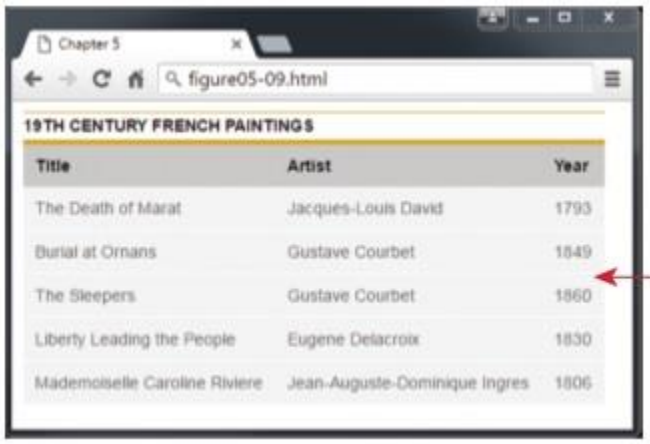

```
table {
   border: solid 1pt black;
   border-spacing: 10pt;
}
td {
   border: solid 1pt black;
}
```




```
caption {
   font-weight: bold;
   padding: 0.25em 0 0.25em 0;
   text-align: left;
   text-transform: uppercase;
   border-top: 1px solid #DCA806;
}
table {
   font-size: 0.8em;
   font-family: Arial, sans-serif;
   border-collapse: collapse;
   border-top: 4px solid #DCA806;
   border-bottom: 1px solid white;
   text-align: left;
}
```

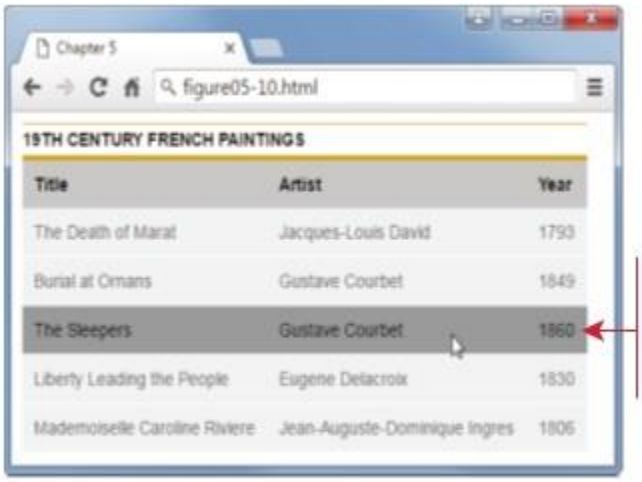



```
thead tr {
   background-color: #CACACA;
}
th {
   padding: 0.75em;
}
```

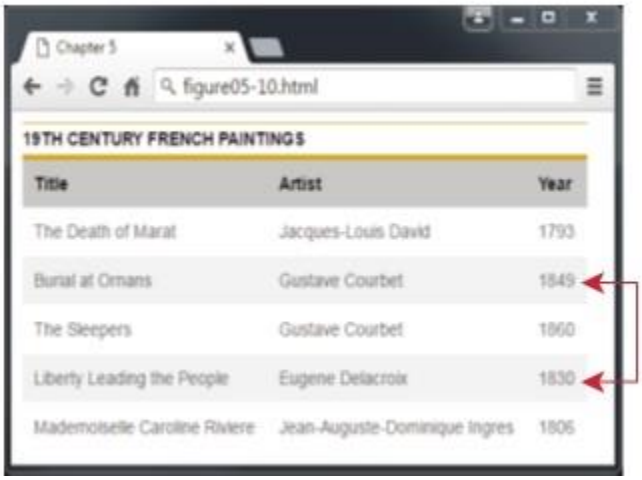



```
tbody tr {
   background-color: #F1F1F1;
   border-bottom: 1px solid white;
   color: #6E6E6E;
}
tbody td {
   padding: 0.75em;
}
```

.




```
tbody tr:hover {
   background-color: #9e9e9e;
   color: black;
}
```




```
tbody tr:nth-child(even) {
   background-color: lightgray;
}
```

Accessible Tables

- Use tables for data, not layout
- Use the <caption> element
- Connect cells with a textual description in the header

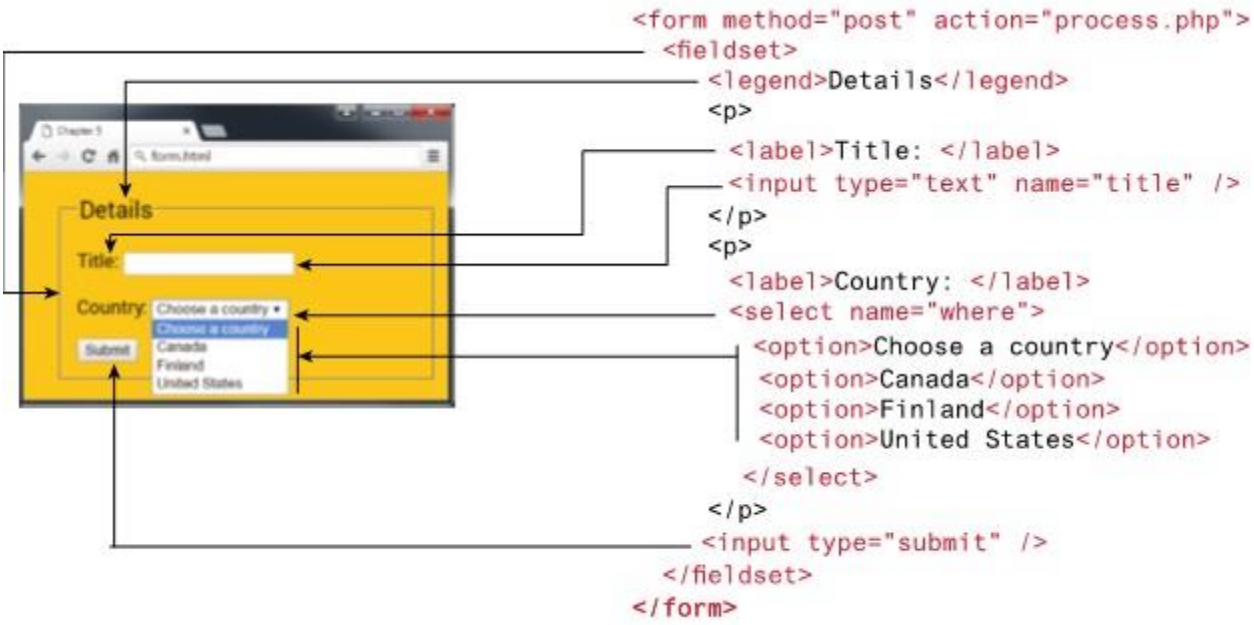
Code:

```
<caption>Famous Paintings</caption>
<tr>
  <th scope="col">Title</th>
  <th scope="col">Artist</th>
  <th scope="col">Year</th>
  <th scope="col">Width</th>
  <th scope="col">Height</th>
</tr>
```

Module 36:

Introducing Forms

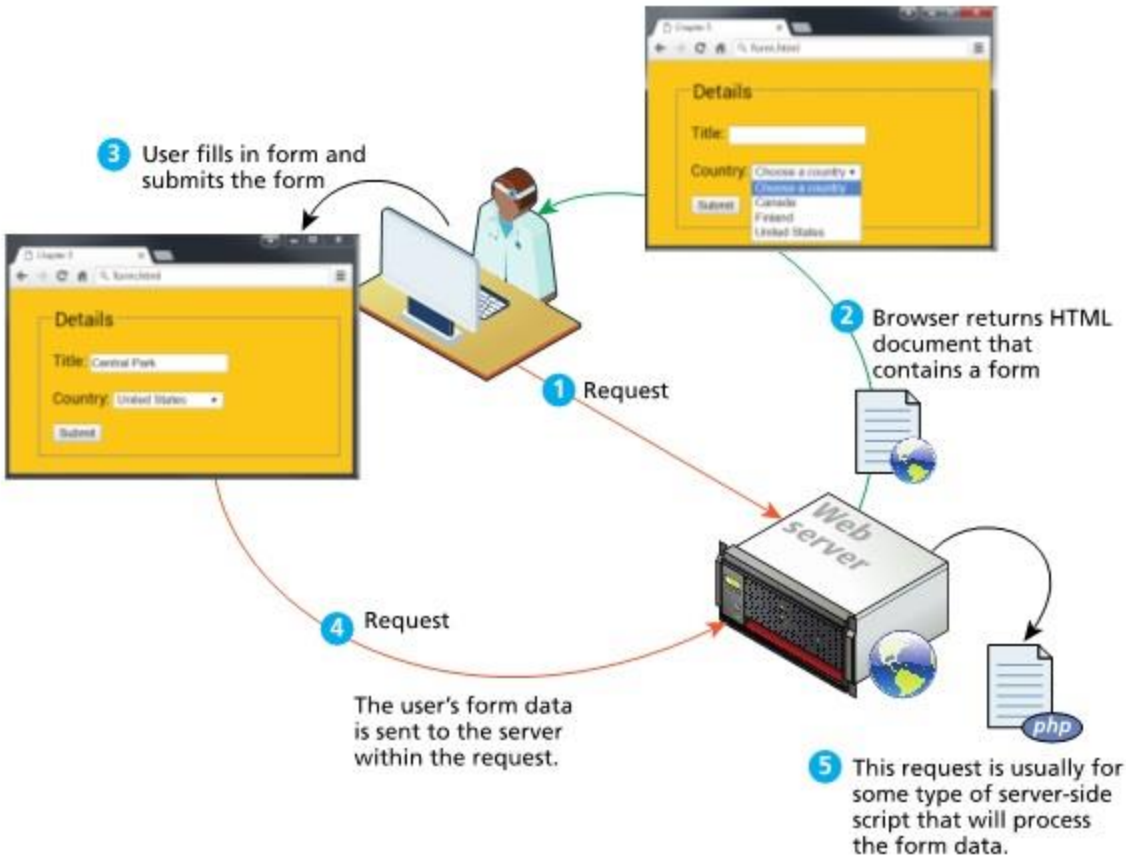
Forms provide the user with an alternative way to interact with a web server. Up to now, clicking hyperlinks was the only mechanism available to the user for communicating with the server.



How Forms Work

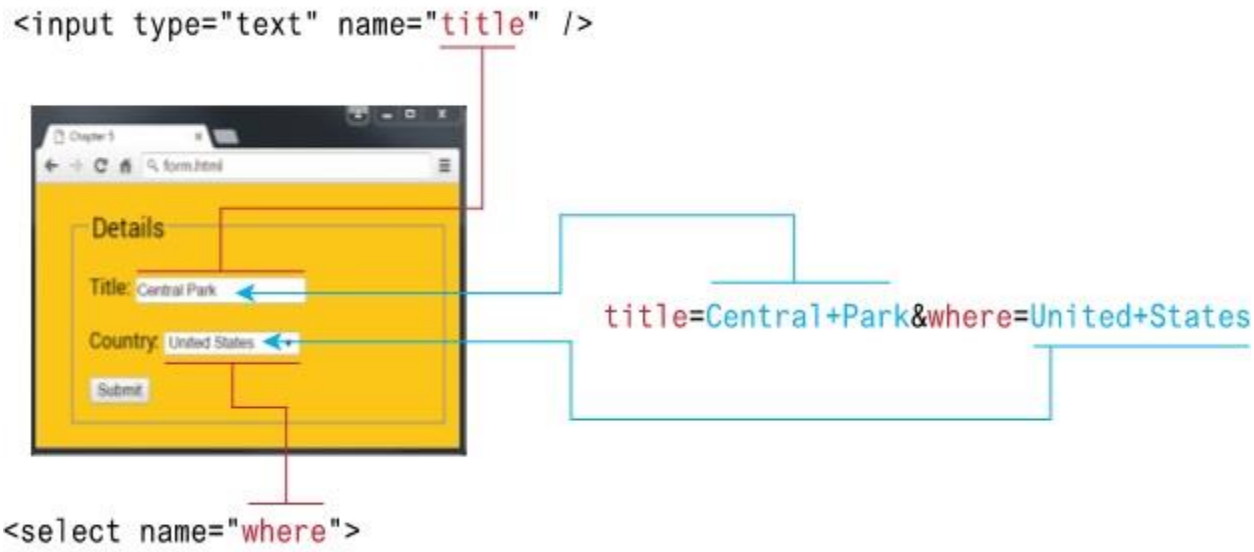
The process begins with a request for an HTML page that contains some type of form on it. This could be something as complex as a user registration form or as simple as a search box. After the user fills out the form, there needs to be some mechanism for submitting the form data back to the server. This is typically achieved via a submit button, but through JavaScript, it is possible

to submit form data using some other type of mechanism.



Query Strings

The browser packages the user's data input into something called a query string. A query string is a series of name=value pairs separated by ampersands (the **&** character).

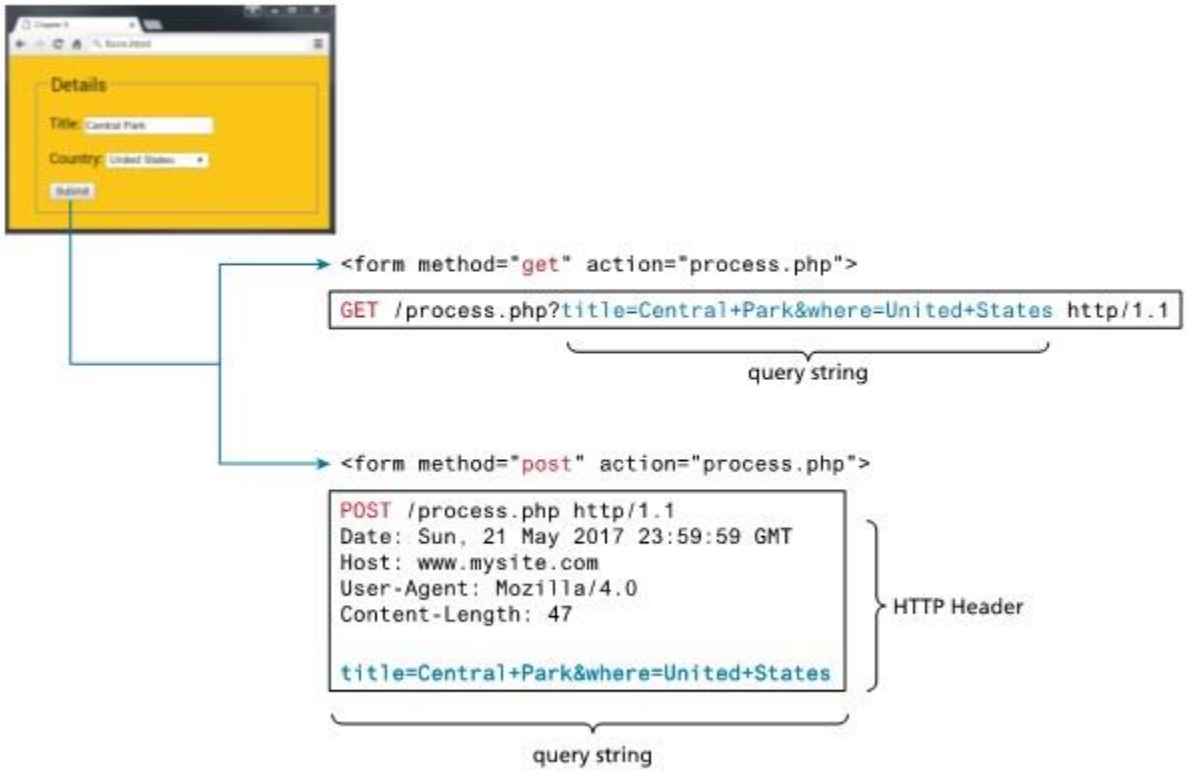


GET and POST

The method attribute specifies how the query string data will be transmitted from the browser to the server. There are two possibilities: GET and POST.

What is the difference between GET and POST?

The difference resides in where the browser locates the user's form input in the subsequent HTTP request. With GET, the browser locates the data in the URL of the request; with POST, the form data is located in the HTTP header after the HTTP variables. Following Figure illustrates how the two methods differ.



GET versus POST

Type	Advantages and Disadvantages
GET	• Data can be clearly seen in the address bar. This may be an • advantage during development but a disadvantage in • production. • Data remains in browser history and cache. Again this may • be beneficial to some users, but a security risk on public • computers.

	• Data can be bookmarked (also an advantage and a • disadvantage). • Limit on the number of characters in the form data • returned.
POST	• Data can contain binary data. • Data is hidden from user. • Submitted data is not stored in cache, history, or • Bookmarks.

Module 37:

Form Control Elements:

Most forms need to gather text information from the user. Whether it is a search box, or a login form, or a user registration form, some type of text input is usually necessary. Different text input controls are mentioned below:

Type	**Description**
Text	Creates a single-line text entry box. <input type="text" name="title" />
textarea	Creates a multiline text entry box. <textarea rows="3" ... />
password	Creates a single-line text entry box for a <input type="password" ... />
search	Creates a single-line text entry box suitable for a search string. <input type="search" ... />
email	Creates a single-line text entry box suitable for entering an email address. <input type="email" ... />
tel	Creates a single-line text entry box suitable for entering a telephone. <input type="tel" ... />
url	Creates a single-line text entry box suitable for entering a URL. <input type="url" ... />

Datalist:

<datalist> element is new addition to HTML5. This element allows to define a list of elements that can appear in a drop-down autocomplete style list for a text element.

Choice Controls:

Forms often need the user to select an option from a group of choices. HTML provides several ways to do this.

1. **Select Lists:**

The <select> element is used to create a multiline box for selecting one or more items. The options can be hidden in a dropdown list or multiple rows of the list can be visible. Option items can be grouped together via the <optgroup> element.

2. **Radio Buttons:**

Radio buttons are useful when you want the user to select a single item from a small list of choices and you want all the choices to be visible.

3. **Check Boxes:**

Checkboxes are used for getting yes/no or on/off responses from the user.

Button Controls:

HTML defines several different types of buttons which are mentioned below:

Type	**Description**
Submit	Creates a button that submits the form data to the server.
Reset	Creates a button that clears any of the user's already entered form data.
button	Creates a custom button. This button may require JavaScript for it to actually perform any action.
image	Creates a custom submit button that uses an image for its display.

Module 38:

Form Control Elements:

Date & Time Controls:

Types	Description
Date	Creates a general date input control. The format for the date is “yyyy-mm-dd.”
Time	Creates a time input control. The format for the time is “HH:MM:SS,” for hours:minutes:seconds.
month	Creates a control in which the user can enter a month in a year. The format is “yyyy-mm.”
week	Creates a control in which the user can specify a week in a year. The format is “yyyy-W##.”

Module 39:

Form Control Elements:

Date & Time Controls:

Types	Description
Date	Creates a general date input control. The format for the date is “yyyy-mm-dd.”
Time	Creates a time input control. The format for the time is “HH:MM:SS,” for hours:minutes:seconds.
month	Creates a control in which the user can enter a month in a year. The format is “yyyy-mm.”
week	Creates a control in which the user can specify a week in a year. The format is “yyyy-W##.”

Module 40:

Image Formats for the Web:

When you see text and images on your desktop monitor or your mobile screen, you are seeing many small squares of colored light called pixels that are arranged in a two-dimensional grid. There are two basic categories of digital representations for images: raster and vector. In a **raster image** (also called a bitmap image) the smaller components are pixels. That is, the image is broken down into a two-dimensional grid of colored squares. Each colored square uses a number that represents its color value. A raster image has a set number of pixels, dramatically increasing or decreasing its size can dramatically affect its quality.
A **vector image** is not composed of pixels but instead is composed of objects such as lines, circles, Bezier curves, and polygons.

File Formats:

1. **JPEG:**

JPEG (Joint Photographic Experts Group) or JPG is a 24-bit, true-color file format that is ideal for photographic images. It uses a sophisticated compression scheme that can dramatically reduce the file size of the image. JPEG is the ideal file format for photographs and other continuous-tone images such as paintings and grayscale images.

2. **GIF:**

The GIF (Graphic Interchange Format) file was the first image format supported by the earliest web browsers. Unlike the 24-bit JPEG format, GIF is an 8-bit or less format, meaning that it can contain no more than 256 colors. GIF files use a much simpler compression system that is lossless, which means that no pixel information is lost.

3. **PNG:**

The PNG (Portable Network Graphics) format is a more recent format. Its main features are:

v. Lossless compression.

v. 8-bit (or 1-bit, 2-bit, and 4-bit) indexed color as well as full 24-bit true color

v. From 1 to 8 bits of transparency.

For normal photographs, JPEG is generally still a better choice because the file size will be smaller than using PNG.

4. **SVG:**

The SVG (Scalable Vector Graphics) file format is a vector format. SVG graphics do not lose quality when enlarged or reduced.

Other Formats:

There are many other file formats for graphical information. The TIF (Tagged Image File) format is a cross-platform lossless image format that supports multiple color depths, 8-bit transparency, layers and color channels, the CMYK and RGB color space, and other features especially useful to print professionals.

Module 41:

Normal Flow:

To understand CSS positioning and layout, it is essential to first understand the idea of normal flow, and how the browser will normally display block-level elements and inline elements from left to right and from top to bottom.

Block-level elements: such as <p>, , <h2>, <ul>, and <table> are each contained on their own line. Because block-level elements begin with a line break, without styling, two block-level elements can't exist on the same line. Block-level elements use the normal CSS box model, in that they have margins, paddings, background colors, and borders.

Inline elements: do not form their own blocks but instead are displayed within lines. Normal text in an HTML document is inline, as are elements such as , <a>, <img>, and .

Module 42:

Positioning Elements:

It is possible to move an item from its regular position in the normal flow. The position property is used to specify the type of positioning, and the possible values. The left, right, top, and bottom properties are used to indicate the distance the element will move; the effect of these properties varies depending upon the position property.

Relative positioning:

In relative positioning an element is displaced out of its normal flow position and moved relative to where it would have been placed.

Static:

The element is positioned according to the normal flow. This is the default.

Fixed:

The element is fixed in a specific position in the window even when the document is scrolled.

Absolute:

The element is removed from normal flow and positioned in relation to its nearest positioned ancestor.

Module 43:

Absolute Positioning:

When an element is positioned absolutely, it is removed completely from normal flow. Thus, unlike with relative positioning, space is not left for the moved element, as it is no longer in the normal flow. Its position is moved in relation to its container block.

Module 44:

Z- Index:

Each positioned element has a stacking order defined by the z-index property (named for the z-axis). Items closest to the viewer (and thus on the top) have a larger z-index value, which can be seen in the first. Unfortunately, working with z-index can be tricky and seemingly counterintuitive.

Module 45:

Fixed Position:

The fixed position value is used relatively infrequently. It is a type of absolute positioning, except that the positioning values are in relation to the viewport (i.e., to the browser window). Elements with fixed positioning do not move when the user scrolls up or down the page. The fixed position is most commonly used to ensure that navigation elements or advertisements are always visible.

Module 46:

Floating Elements:

It is possible to displace an element out of its position in the normal flow via the CSS float property. An element can be floated to the left or floated to the right. When an item is floated, it is moved all the way to the far left or far right of its containing block and the rest of the content is “re-flowed” around the floated element.

Floating within a Container:

Floated item moves to the left or right of its container (also called its containing block).

Floating Multiple Items:

One of the more common usages of floats is to place multiple items side by side on the same line. When you float multiple items that are in proximity, each floated item in the container will be nestled up beside the previously floated item. All other content in the containing block (including other floated elements) will flow around all the floated elements.

Clear Property:

You can stop elements from flowing around a floated element by using the clear property. Values for clear property are shown below:

Values	Description
Left	The left-hand edge of the element cannot be adjacent to another element.
Right	The right-hand edge of the element cannot be adjacent to another element.
Both	Both the left-hand and right-hand edges of the element cannot be adjacent to another element.
None	The element can be adjacent to other elements.

Containing Floats:

Another problem that can occur with floats is when an element is floated within a containing block that contains only floated content. In such a case, the containing block essentially disappears.

Module 47:

Overlaying and Hiding Elements:

One of the more common design tasks with CSS is to place two elements on top of each other, or to selectively hide and display elements. Positioning is important to both of these tasks. Positioning is often used for smaller design changes, such as moving items relative to other elements within a container. In such a case, relative positioning is used to create the positioning context for a subsequent absolute positioning move.

There are in fact two different ways to hide elements in CSS: using the display property and using the visibility property. The **display property** takes an item out of the flow: it is as if the element no longer exists. The **visibility property** just hides the element, but the space for that element remains.

Module 48:

Constructing Multicolumn Layout (using Float):

A typical layout may very well use both positioning and floats. There is unfortunately no simple and easy way to create robust multicolumn page layouts.

Using Floats to Create Columns:

Using floats is perhaps the most common way to create columns of content. The first step is to float the content container that will be on the left-hand side. Remember that the floated container needs to have a width specified. The background of the floated element stops when its content ends. If we wanted the background color to descend down to the footer, then it is difficult (but not impossible) to achieve this visual effect with floats.

Three- Column Example:

Another approach for creating a three-column layout is to float elements within a container element. This approach is actually a little less brittle because the floated elements within a container are independent of elements outside the container. The floated content must appear in the source before the nonfloated content. This is the main problem with the floated approach: that we can't necessarily put the source in an SEO-optimized order.

Using Positioning to Create Columns:

Positioning can also be used to create a multicolumn layout. This approach typically uses some type of container that establishes the positioning context. With positioning it is easier to construct our source document with content in a more SEO-friendly manner; in this case, the main can be placed first.

Problem with Absolute Positioning:

When an item is positioned, it is removed entirely from normal flow, so subsequent items will have no "knowledge" of the positioned item. One solution to this type of problem is to place the footer within the main container.

Module 49:

Constructing Multicolumn Layout using flexbox:

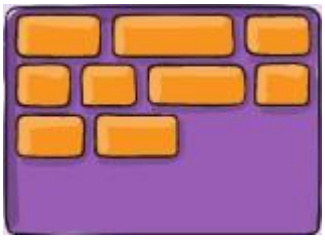
Flex layout is to give the container the ability to alter its items' width/height to best fill the available space.

Flexbox parent properties:

1. **flex-start**:

Items are packed toward the start of the flex-direction.

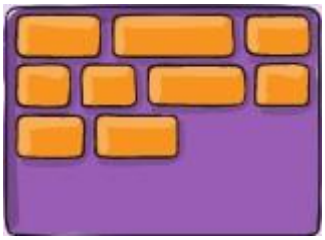
Example:



2. flex-end:

Items are packed toward the end of the flex-direction.

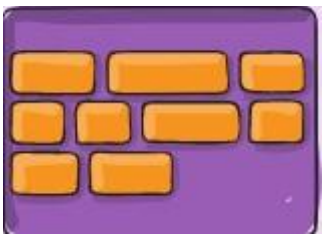
Example:



3. Center:

Items are centered along the line.

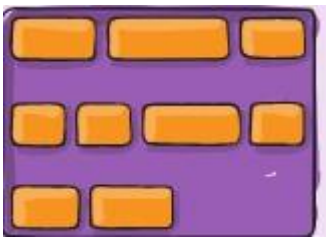
Example:



4. Space-between:

Items are evenly distributed in the line; first item is on the start line, last item on the end line.

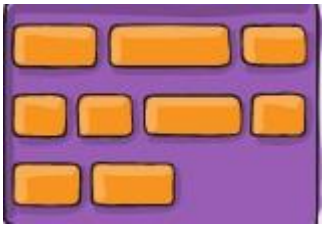
Example:



5. space-around:

Items are evenly distributed in the line with equal space around them.

Example:



Flexbox Child properties:

1. **Flex-grow:**

It tell us what amount of the available space inside the flex container the item should take up. By default we have equal space for flex grow and it set to 1, this means container will be distributed equally to all children.

Example:



Flexgrow set to be 1

If one of the children has a value of 2, that child would take up twice as much of the space either one of the others.

Example:



Flexgrow set to be 2

2. Flex-basis:

This defines how much space an element is taking initially.

3. Flex-shrink

This defines the ability for a flex item to shrink if necessary.

Module 50:

Approaches to CSS Layout:

One of the main problems faced by web designers is that the size of the screen used to view the page can vary quite a bit. Users with the large monitor might expect a site to take advantage of the extra size; users with the small monitor will expect the site to scale to the smaller size and still be usable. Satisfying both users can be difficult; the approach to take for one type of site content might not work as well with another site with different content.

Fixed Layout:

The first approach is to use a fixed layout. In a fixed layout, the basic width of the design is set by the designer, typically corresponding to an “ideal” width based on a “typical” monitor resolution. Fixed layouts are created using pixel units, typically with the entire content within a container (often named "container", "main", or "wrapper") whose width property has been set to some width. The advantage of a fixed layout is that it is easier to produce and generally has a predictable visual result.

Fixed layouts have other drawbacks. For larger screens, there may be an excessive amount of blank space to the left and/or right of the content. Much worse is when the browser window shrinks below the fixed width; the user will have to horizontally scroll to see all the content.

Liquid Layout:

The second approach to dealing with the problem of multiple screen sizes is to use a liquid layout (also called a fluid layout). In this approach, widths are not specified using pixels, but percentage values. The obvious advantage of a liquid layout is that it adapts to different browser sizes, so there is neither wasted white space nor any need for horizontal scrolling.