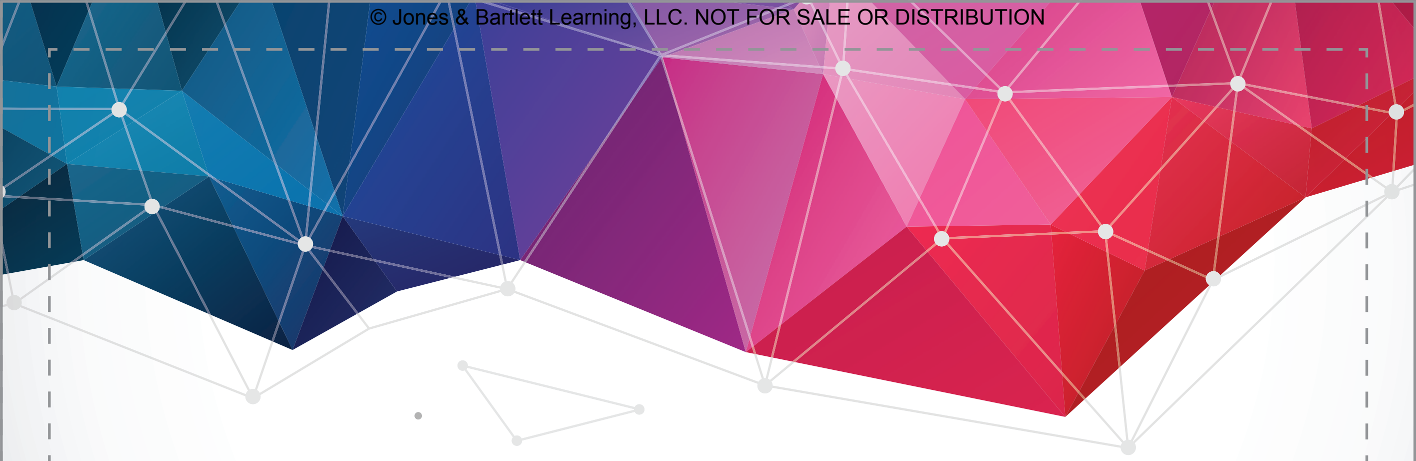


CHAPTER ONE Setting the Scene

CHAPTER CONTENTS

Overview and Objectives

1.1 What Is the Internet?
1.2 What Is the World Wide Web?
1.3 What Is Meant by a Client-Server Architecture?
1.4 How Do Web Browsers and Web Servers Fit the Client-Server Model?
1.5 How Do Web Browsers and Web Servers Communicate?
1.6 A Real-World E-Commerce Website
1.7 The Technologies We Will Discuss
1.8 Some Alternative Technologies and Additional Tools

Summary

Quick Questions to Test Your Basic Knowledge
Short Exercises to Improve Your Basic Understanding
Exercises on the Parallel Project
What Else You May Want or Need to Know
References

Overview and Objectives

While the *Internet* has been around for more than four decades, its usage has exploded since the advent of the *World Wide Web* in the early 1990s. From a web developer's point of view, the web technologies of that time were very simple, mostly consisting of *HyperText Markup Language* (*HTML*) in its most basic form. Almost everything that one needed to know could be explained in a book of modest size. The technologies have since proliferated beyond anyone's expectations, and it is now impossible for any individual to be an expert in everything that is used in website development around the world.

In this book we will make an attempt to capture the essential features of several technologies one can use to launch a career as a web developer. The aim is to provide sufficient information about web development without overwhelming the reader. We will use the example of an e-commerce business and demonstrate that the core knowledge contained in this book can indeed be used to develop credible websites.

At the end of this chapter, we list the various technologies that will be covered in this book, and also mention a few alternate competing technologies that we will not discuss, just to give you a sense of what is "out there". There are many libraries, frameworks, and other tools that a web developer can use to help build or enhance a website, but that we will have neither the time nor the space to consider. This is a text about developing a reasonable website "from scratch", and giving you, our reader, the background skills necessary to understand, tweak, and even extend those other more sophisticated tools if and when you decide to use them.

Beginning with this one, each chapter of this text ends with a collection of activities to be performed, and other follow-up material. The first of these sections contains "short-answer questions to test your basic knowledge" (of the material directly covered in the current chapter). Next comes a section of "short exercises to improve your basic understanding" (of the procedures discussed in the current chapter). These exercises will be mostly of the hands-on variety.

Then there is the key section covering "exercises on the parallel project". This will be an ongoing project throughout the text, which will allow you to develop a website whose content will be of your own choosing, but whose functionality will parallel that of the major **Nature's Source** online business example that we introduce in the text and continue to develop and enhance with new capabilities from chapter to chapter.

There is also a section on "what else you may want or need to know". This section provides additional material, however brief, on some topics that were not covered, or perhaps merely mentioned, in the chapter itself, and either directly supplements that material or suggests follow-up action that you might take to extend and consolidate your knowledge. Pursuing these additional resources can help you to achieve a higher level of sophistication in the use of those technologies described in the chapter, or to explore some of the alternate competing technologies. Time constraints may not permit you to complete all of the suggested activities in this "what else" group, but we hope to provide sufficient direction so that the book will not only help you ramp up to the information superhighway, but also give you pointers on going the distance if you wish to do so.

A final section on references will contain annotated links to websites of interest, where you are most likely to find the most up-to-date information on any particular topic. Keep in mind that links may change or disappear completely, but if you enter relevant key words into a search engine like Google, you will likely be able to find where the link went, or discover a suitable alternative site.

Our goals in this chapter are the following:

- To distinguish between the Internet and the World Wide Web
- To explain client-server architectures, as illustrated by web browsers and web servers
- To discuss how web browsers and web servers communicate
- To take a brief look at a real-world e-commerce website
- To outline the technologies we will discuss in this text, and mention a few of the competing technologies and other tools that we will not discuss

1.1 What Is the Internet?

The *Internet* is a worldwide collection of computers and other devices connected by various means such as copper wire, fiber optics, and wireless communications of various kinds. Businesses, governments, organizations, schools and universities, private homes, and "people on the go" are all "connected to the Internet", or "wired".

The Internet came into existence at the end of the 1960s, driven by a cold war desire of the U.S. military to have a secure means of communication in the event of nuclear war. Initial development was under the auspices of the Advanced Research Projects Agency (ARPA) in the United States, and what eventually became "The Internet" was at first called ARPANET. The first developments were performed by a small number of research institutions funded by ARPA, but by the late 1970s and early 1980s several other networks had been developed. With the growing interest of businesses and private individuals, over one million computers had been connected by 1992. The Internet has continued to expand at rates that are difficult to measure, but it has clearly become ubiquitous, in much of the "developed" world at least.

1.2 What Is the World Wide Web?

The *World Wide Web* (generally abbreviated as *WWW* or *W3*) is a "software infrastructure" consisting of many different communication standards for gaining access to, and exchanging information over, the Internet. Many different kinds of computer software applications that run on computers connected to the Internet use those communication standards to provide that access and/or make those exchanges. The presence of a *web browser* is the most familiar sign that access to the World Wide Web, or just "the web", is available.

The development of what eventually became the World Wide Web was started in the late 1980s by Tim Berners-Lee and others at CERN (the French acronym for the European Laboratory for Nuclear Research). The idea was to use the notion of *hypertext* so that scientific documents could be made available over the Internet to anyone who had a connected computer. The term *hypertext* refers to the "linking" of documents to one another in such a way that one can easily go from viewing one document to viewing another related document via a "link" to that other document that appears in the first document. *HTML* was developed for the purpose of describing the structure of documents containing such links that would be made available, and "browsers" with simple text-based interfaces (Lynx being one of the better known ones) were used to retrieve and display these documents. It was not until Mosaic, the first widely used browser with a *Graphical User Interface* (*GUI*), was developed that the World Wide Web really took off. The rest, as everyone now knows, is history.

1.3 What Is Meant by a Client-Server Architecture?

The *client-server architecture* is one approach to communication between two software applications that usually (but not always) reside on physically distinct machines. In typical client-server communication a *client machine* first sends a request to a *server machine*. The server then either honors the request by returning to the client whatever was requested, or returns an error that indicates why the request could not be honored. At least this is the ideal response, and when an error is returned it is up to the software (and perhaps a user) on the client side to decide what happens next.

1.4 How Do Web Browsers and Web Servers Fit the Client-Server Model?

In the client-server context of the web, a "web browser" is a "client program" that a user employs to contact, over the Internet, other computers that are running "server software" and are therefore capable of responding to requests sent by a browser for information to be displayed in the browser window. At the time of this writing (2015) Google's Chrome web browser appears to hold the dominant market share, with Firefox a strong second. The number of Microsoft

Internet Explorer users has fallen rapidly in recent years, at least partly because of its failure to implement many web standards properly, but at the moment it is still in third place. Opera is an excellent browser, but still has a relatively small number of users, and Safari is often the browser of choice among Apple aficionados. This browser scene is likely to exhibit even more volatility as time goes on, particularly if Microsoft puts a significant effort into regaining lost user share.

The term *web server* is potentially confusing because sometimes it refers to a software program, and sometimes it refers to the computer on which that program runs. Just knowing this should help you to tell from the context which one is being referenced in any given situation. In either case, the "web server" is the server side of the client-server architecture in the context of the web, and it is the program (or the machine) that responds to requests from browsers. At the time of this writing the most popular web server is the open-source Apache, but Microsoft's IIS (Internet Information Server) is widely used by those in the Microsoft camp.

1.5 How Do Web Browsers and Web Servers Communicate?

In this section, we will look at a number of concepts that are necessary to understand if we are to get a sense of how web browsers and web servers communicate with each other.

1.5.1 Web Protocols and Layered Communication Architectures

Communication protocol

A *communication protocol* is simply an agreement by two or more parties about what rules will be followed when communications between or among the parties take place. Humans use (mostly informal) protocols all the time when communicating. Think, for example, of making a simple telephone call: A caller dials the number, the phone at the other end rings, a person picks up the receiver and says, "Hello", the caller identifies himself or herself and states the purpose of the call, the recipient responds, there are perhaps more exchanges, after which the caller says, "Thank you. Goodbye." Finally, the caller hangs up, and the recipient then hangs up as well. That's a communication protocol in action.

Web protocol

A *web protocol* is, similarly, an agreed-upon set of rules and data formats to be used when two or more computers or other devices, or application programs running on those machines, wish to communicate across the Internet, usually but not always on behalf of human users. In any given communication it is likely that there will be several different protocols involved.

Common web protocols

There are many protocols in use on the web. Here is a very short list of some of the more common ones:

- *TCP/IP* (*Transmission Control Protocol/Internet Protocol*), a two-part protocol that underlies pretty much everything that travels over the web. This is the low-level "lingua franca" of the World Wide Web. If TCP/IP went away tomorrow, the web would cease to exist. One of the reasons it is so widely used is that it guarantees delivery of the information that was sent.
- *UDP* (*User Datagram Protocol*), another protocol that can also be used as the underlying transport protocol for information, and though it may be faster to use if you are moving large multimedia files (for example), it does not guarantee that all of the information will arrive safely. This may not be an important consideration if you do not care that your final photograph is missing a pixel or two.
- *HTTP* (*HyperText Transfer Protocol*), the protocol that browsers use to send requests for information to servers and that a server uses to send the requested information back to a browser.
- *FTP* (*File Transfer Protocol*), the protocol used to transfer files from one computer to another across the Internet.
- *TELNET* (*TELephone NETwork*) and *SSH* (*Secure SHell*) are both protocols that can provide "terminal emulation" when used to connect, over the Internet, to a remote computer and log in to an account on that computer. TELNET has been around for many years, but its use is discouraged these days because of security concerns, in favor of SSH, which is a more secure protocol for the transfer of information.

Layered communication architectures

One difficulty with trying to understand how things happen on the web is that there are so many of these web protocols, and often it is not clear which ones are in play. A second difficulty arises from the fact that all these protocols are just parts of a much "bigger picture".

In a nutshell, any communication over the Internet between two computers can be viewed in the following way: Data starts in an application on the first computer, "trickling down" to the actual hardware on that machine, passing over the Internet to the hardware of the second computer, then "bubbling up" to the application running on that second machine that is expecting it. If the second application replies, the process is reversed. On each machine, the data passes through several communication "layers".

There are different models of these layers, including the seven-layer *Open Systems Interconnect Model* and the four-layer *Internet Model*. Much more could be said about these models, and the many protocols that are found at the various layers within them, but it would take us too far afield. The average web developer does not need to know any more about them than you now know. (But we do suggest you seek out more information on these concepts in the **What Else** section at the end of this chapter.)

1.5.2 Web Addresses and Address Resolution via DNS

Just as a letter being sent by regular mail needs to have the address of its destination affixed if it is not to go astray, so does a request for information sent from a browser out on the Internet need to supply the "address" of the recipient to which the request is being sent.

IP addresses

Every computer attached to the Internet has a unique *IP address*, which has the form a.b.c.d, where each of the values a, b, c, and d is a positive integer in the range 0..255, and the intervening periods are a required part of the syntax. This allows for just over four billion different 32-bit addresses. So, if we are not to exhaust our available supply of addresses, another scheme must eventually be implemented. This is the IPv6 scheme (our current four-part address scheme is called IPv4), which is currently under development and implementation, and which will provide more Internet addresses than we are ever likely to need. Blocks of IP addresses are issued to organizations, businesses, educational institutions, governments, and even countries, which then redistribute them "internally" to subgroups and individuals.

Fully qualified domain names

So, though a computer will find it very convenient to work with a numerical "address" like 123.234.235.236, humans prefer names to numbers since they are usually easier to remember. Thus a computer with the preceding address may also have a name, or, more accurately and completely, a *Fully Qualified Domain Name* (*FQDN*), which could be something of the form

```
myhost.mycompany.myregion.com
```

if it's a commercial website, or

```
someschool.downyonder.edu
```

if it's an educational site.

Host machines and domains

The characters following the last period of an FQDN indicate the largest "domain" to which that name belongs, and can be a country code, such as ca for Canada, or one of a small number of specific designations, such as .edu for an educational institution, .com for a commercial enterprise, .gov for a government, or .org for an organization (essentially noncommercial) of some kind. The name at the left of the domain name (i.e., to the left of the first period) is generally the name of the host machine, and as one proceeds from left to right through the domain name, the succeeding names represent larger and larger domains to which that host machine belongs.

The domain name system and domain name servers

Because humans tend to use FQDNs and computers will use the actual numerical IP address when communicating with one another, there has to be a system to convert addresses from one form to the other. This is the *Domain Name System* (*DNS*), and the machines connected to the Internet that perform the service of "resolving" any FQDN to its corresponding IP address are called *Domain Name Servers* (for which the acronym is also *DNS*), or simply "name servers". Although it is possible to use IP addresses directly when "surfing the Web", few humans would do so, so this process of "address resolution" is a very important part of what goes on as part of the traffic over the web.

1.5.3 URLs, URNs, and URIs

The most frequently encountered of these three acronyms is (probably) URL, then URI, and finally (and much less frequently, if at all) URN, though this may be changing.

Uniform Resource Locator (URL)

A URL is, as its name suggests, a *uniform* (or "standard") way of referring to the *location* of a web document (or, more generally, to the location of a web *resource* of whatever kind, not necessarily what we would normally think of as a "document"). Naturally, therefore, the fully qualified domain name of the host machine on which the resource is located forms an integral part of the URL for that resource. However, it is not enough to know *where* the resource is located. One must also know, and be able to specify, *how* the resource will be accessed (i.e., the method, or *protocol*, that will be used to access the resource). After all, not every web "resource" is just a page to be displayed. So, a URL quite often has the form

```
scheme:address_of_resource
```

in which `scheme` is, more often than not, the familiar `http` (though it could very well be something else, such as `ftp`), and `address_of_resource` itself has the following form:

```
//FQDN/path_from_document_root/name_of_resource
```

Thus if you enter something very typical, like

```
http://cs.smu.ca/jobs/2015/current.html
```

into your browser's "address window" and click on Go or press Enter, you are saying you want to retrieve the document current.html using the http protocol scheme, from the server whose fully qualified domain name is cs.smu.ca. The forward slash (/) immediately following cs.smu.ca refers to the directory on that host, which is the host's *document root* (the main directory where the web server stores files that it can serve to requesting browsers). This directory may, of course, also have lots of subdirectories that also contain "servable" files, and which help to keep the website organized. The rest of the path (starting from the document root, wherever it might be located, which is something we cannot determine from the URL) is jobs/2015/, and the final item is the name of the actual desired resource, current.html. Sometimes you will see a URL that looks like

```
http://cs.smu.ca/~porter/courses/index.html
```

in which the *tilde* symbol (~) indicates that the name porter is the name of a user on the host system and that cs.smu.ca/~porter/ is the home directory (or personal document root) for this particular user on this server.

Uniform Resource Name (URN)

A URN is a name that has the same form as a URL, but may not identify an actual location on the Internet. So, a URN can be used to talk about something without implying its existence or indicating how to retrieve a particular resource referenced by the name.

Uniform Resource Identifier (URI)

A URI is a more general concept than either a URL or a URN. According to Wikipedia,[1] the "contemporary" viewpoint is that URLs and URNs are both "context-dependent aspects of a URI and rarely need to be distinguished". In fact, it is suggested in the same Wikipedia article that the term URL may be falling into disuse, since it is "rarely necessary to distinguish between URIs and URLs", and the more "user-friendly" term *web address* is now more frequent in any case.

1.6 A Real-World E-Commerce Website

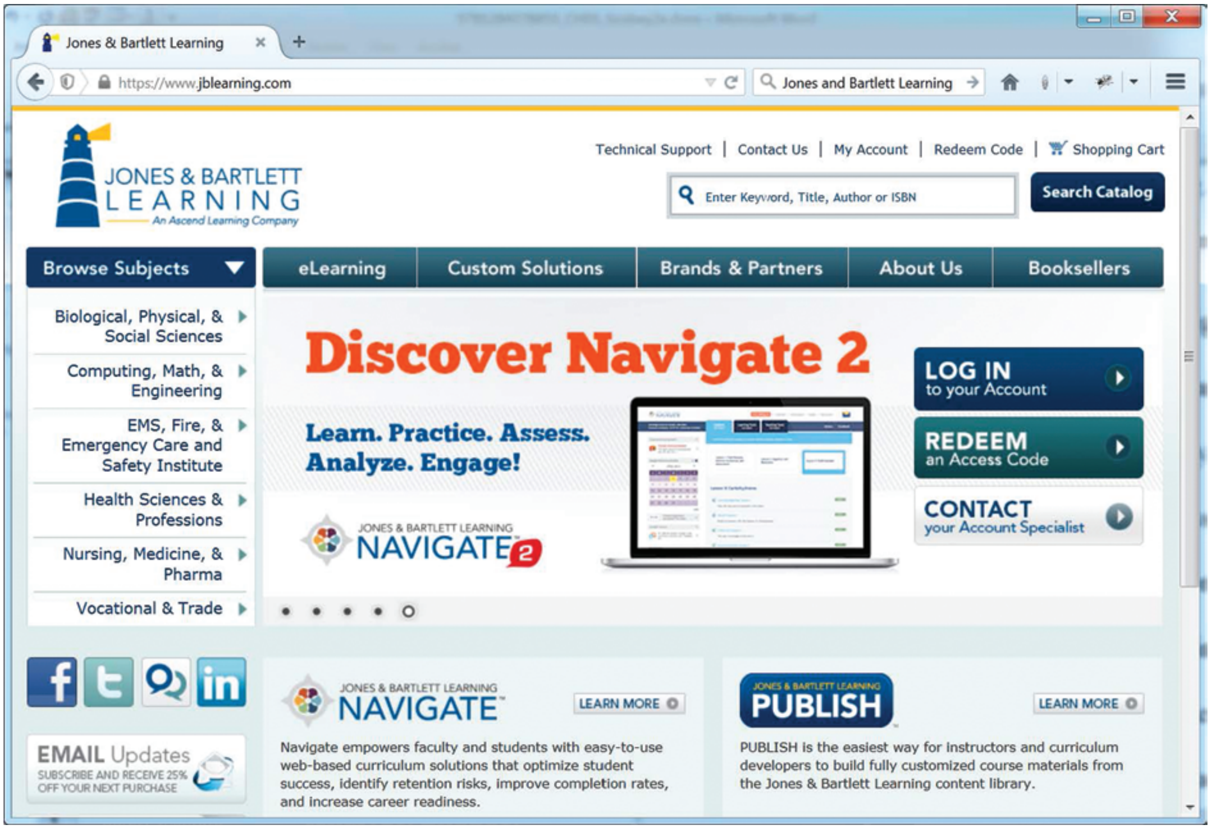
The bold prediction from the 1990s that "If your business is not on the World Wide Web, you will soon not be in business" still contains a certain ring of truth today. It is true that the lack of a web presence may not drive a small business with a loyal clientele and ongoing word-of-mouth advertising out of business. However, many such businesses have established, and continue to establish, a web presence to communicate information to their existing clients and attract new customers. This book will discuss technologies that can be used to add increasing levels of sophistication to the web presence of a business.

Let us look at the well-developed e-commerce website of Jones & Bartlett Learning, the publisher of this book, to understand some typical aspects of an e-commerce website.

FIGURE 1.1 shows a partial view of the home page of Jones & Bartlett Learning. The top-left corner has the logo of the company and at the top-right is a "search window" that allows the user to search for items of interest on the website.

---

[1] See http://en.wikipedia.org/wiki/Uniform_Resource_Identifier.


Jones & Bartlett Learning
https://www.jblearning.com
Jones and Bartlett Learning
Technical Support | Contact Us | My Account | Redeem Code | Shopping Cart
JONES & BARTLETT LEARNING
An Ascend Learning Company
Enter Keyword, Title, Author or ISBN
Search Catalog
Browse Subjects
eLearning
Custom Solutions
Brands & Partners
About Us
Booksellers
Biological, Physical, & Social Sciences
Computing, Math, & Engineering
EMS, Fire, & Emergency Care and Safety Institute
Health Sciences & Professions
Nursing, Medicine, & Pharma
Vocational & Trade
Discover Navigate 2
Learn. Practice. Assess. Analyze. Engage!
JONES & BARTLETT LEARNING NAVIGATE 2
LOG IN to your Account
REDEEM an Access Code
CONTACT your Account Specialist
EMAIL Updates
SUBSCRIBE AND RECEIVE 25% OFF YOUR NEXT PURCHASE
JONES & BARTLETT LEARNING NAVIGATE
LEARN MORE
JONES & BARTLETT LEARNING PUBLISH
LEARN MORE

Navigate empowers faculty and students with easy-to-use web-based curriculum solutions that optimize student success, identify retention risks, improve completion rates, and increase career readiness.

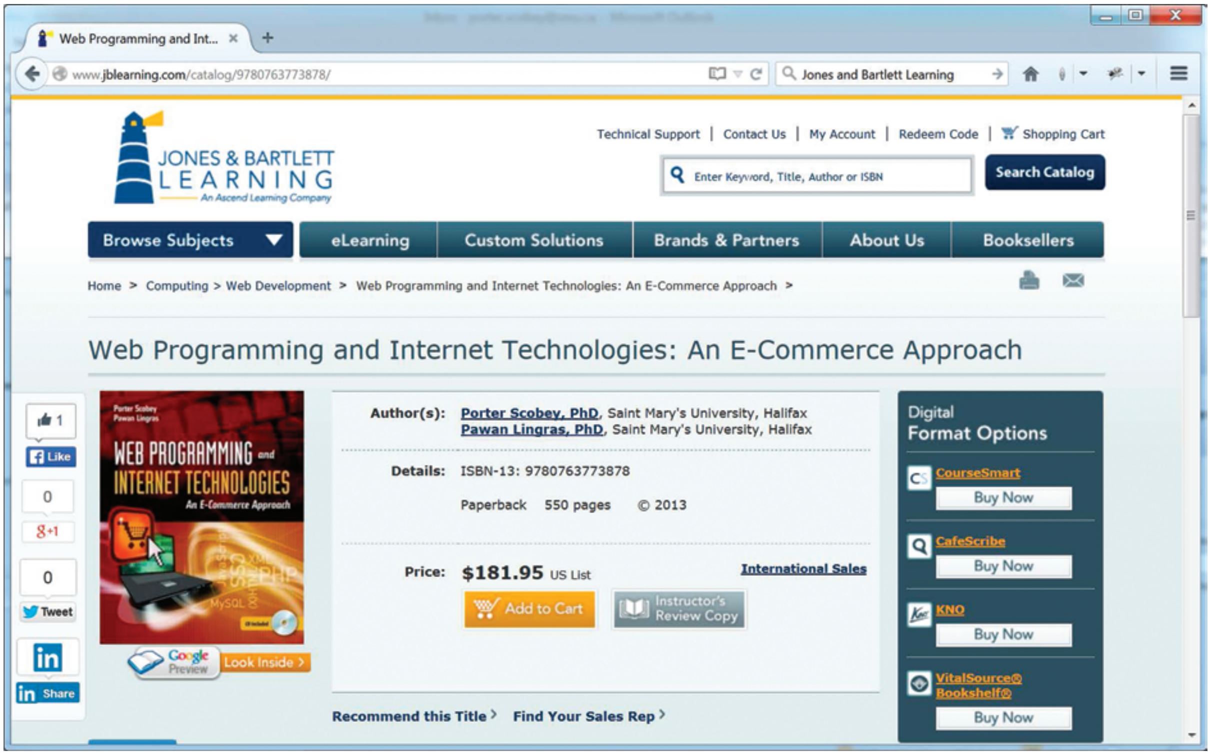
PUBLISH is the easiest way for instructors and curriculum developers to build fully customized course materials from the Jones & Bartlett Learning content library.

FIGURE 1.1 `graphics/ch01/jbHome.jpg`
The home page of Jones & Bartlett Learning (publisher of this text) at `http://www.jblearning.com/`.

A menu of various options appears on the toolbar below the logo, and an additional menu of subject areas appears down the left side. These types of menu options are typical of most commercial websites, though they may appear in many different forms. In general they allow you to get more information about a company, contact the company, shop for its products, and obtain help, as they do in this case. Here the rest of the page includes information about the products and links to obtain more information about them.

The structure of such a web page is specified using the HTML, and its layout and styling with Cascading Style Sheets (CSS). We will be studying the essential features of HTML in Chapter 3, and CSS in Chapter 4.

If you choose different menu options, you will notice that the basic layout of the site remains consistent as you move from one part of the website to another. Such a consistent presentation is also best achieved by the use of CSS. If you hover your mouse over any one of the subject-area menu options at the left, you can access a sequence of "popup" submenu items that will permit you to "drill down" deeper into the website to get to more and more specific information. Such



Web Programming and Int...

www.jblearning.com/catalog/9780763773878/

Jones and Bartlett Learning

JONES & BARTLETT LEARNING An Ascend Learning Company

Technical Support | Contact Us | My Account | Redeem Code | Shopping Cart

Enter Keyword, Title, Author or ISBN — Search Catalog

Browse Subjects | eLearning | Custom Solutions | Brands & Partners | About Us | Booksellers

Home > Computing > Web Development > Web Programming and Internet Technologies: An E-Commerce Approach >

Web Programming and Internet Technologies: An E-Commerce Approach

Author(s): Porter Scobey, PhD, Saint Mary's University, Halifax
Pawan Lingras, PhD, Saint Mary's University, Halifax

Details: ISBN-13: 9780763773878

Paperback 550 pages © 2013

Price: $181.95 US List — International Sales

Add to Cart — Instructor's Review Copy

Google Preview — Look Inside >

Recommend this Title > Find Your Sales Rep >

Digital Format Options

CourseSmart — Buy Now

CafeScribe — Buy Now

KNO — Buy Now

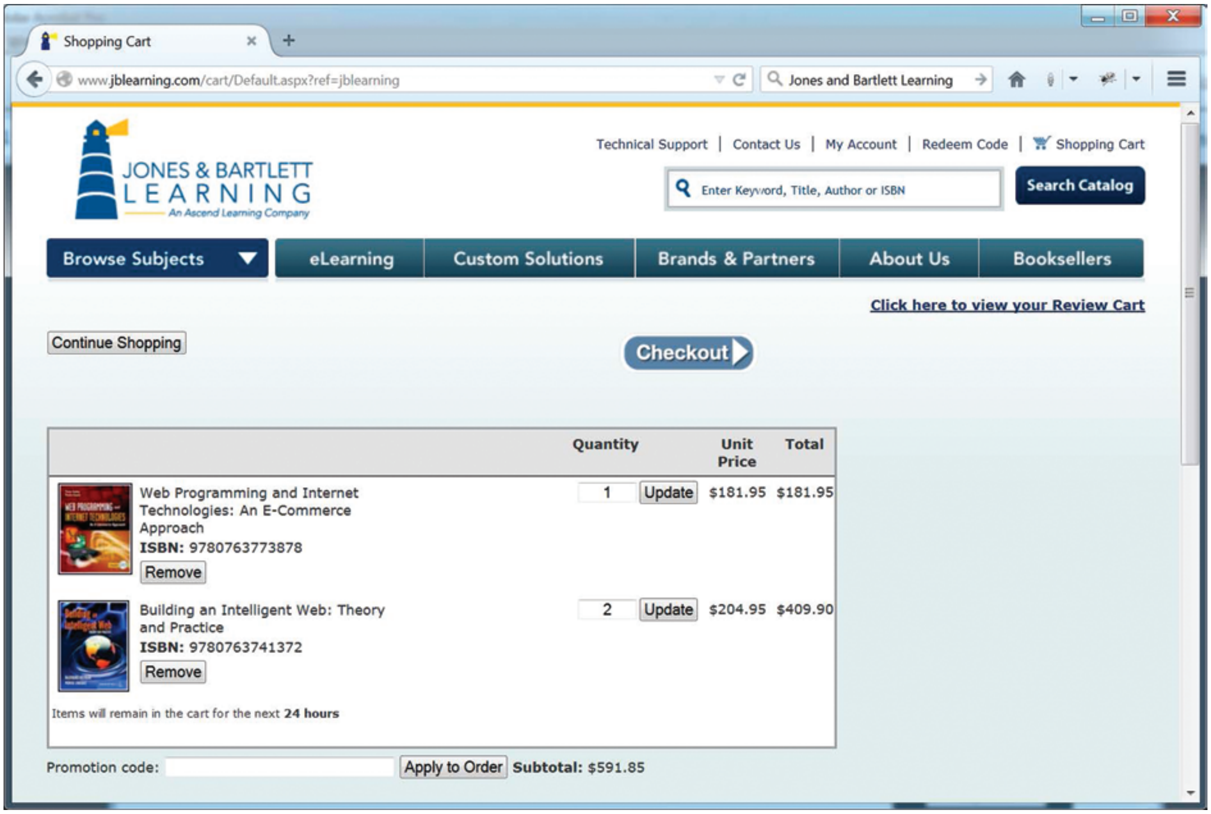
VitalSource® Bookshelf® — Buy Now

FIGURE 1.2 `graphics/ch01/jbBook.jpg`
A web page display showing one of many book titles from Jones & Bartlett Learning.

dynamic behavior is often achieved with JavaScript, the topic of Chapters 6 and 7. JavaScript will also allow us to provide simple animations rotating business-related images on our home page. This provides an effect that is similar to, though simpler than, that of the alternating display shown on the Jones & Bartlett home page.

If we click on any one of the subject-matter category options and choose a book title, we will be entering the realm of e-commerce, as shown in **FIGURE 1.2**, where we see the first edition of this text as an example. Not only can we get detailed information about the book, we can also add it to our "shopping cart".

If we decided to buy multiple titles, our "order form" would appear similar to that shown in **FIGURE 1.3**, where we have decided to purchase one copy of the first edition of this book and two copies of another book (also coauthored by one of the authors of this text). This web page is essentially an online "order form", which allows us to enter the desired quantity of each book. We introduce HTML forms in Chapter 5 and discuss the validation and processing of form data in Chapters 6, 7, and 10. A "shopping cart" order form like this is dynamically generated using web programming with languages such as Personal Home Page (PHP). Note that the information used to create this dynamic form comes both from the user and from a database that is internal to the


Shopping Cart
www.jblearning.com/cart/Default.aspx?ref=jblearning
Jones and Bartlett Learning
JONES & BARTLETT LEARNING
An Ascend Learning Company
Technical Support | Contact Us | My Account | Redeem Code | Shopping Cart
Enter Keyword, Title, Author or ISBN
Search Catalog
Browse Subjects
eLearning
Custom Solutions
Brands & Partners
About Us
Booksellers
Click here to view your Review Cart
Continue Shopping
Checkout

	Quantity		Unit Price	Total
Web Programming and Internet Technologies: An E-Commerce Approach **ISBN:** 9780763773878 Remove	1	Update	$181.95	$181.95
Building an Intelligent Web: Theory and Practice **ISBN:** 9780763741372 Remove	2	Update	$204.95	$409.90

Items will remain in the cart for the next **24 hours**

Promotion code: Apply to Order **Subtotal:** $591.85

FIGURE 1.3 `graphics/ch01/jbCheckout.jpg`
An order form page from the Jones & Bartlett Learning site showing the purchase of two titles.

Jones & Bartlett Learning website. We will see how PHP and a popular database package called MySQL can be used to handle such shopping cart activity for our fictional business in Chapters 9 and 10.

1.7 The Technologies We Will Discuss

Our goal in this text is to familiarize you with most of the basic principles underlying website development. At the same time, you need to be aware that the implementation of those principles on an actual website can be performed in various ways, and we will have the time and space to introduce only some of the implementation technologies.

Our approach is to take the *open-source* route, which means that we shall have the distinct advantage of being able to use a lot of "production quality" free software applications and utilities. This kind of software should never be regarded as inferior for the simple reason that it is free. In fact, software often tends to be one of the great counterexamples to the old saying, "You get what you pay for." With open-source software, history shows that you often get much more than you pay for. And the hard truth is that you may get bad software no matter how much you pay for it. As always, the safest rule to follow is: Buyer beware! This applies, of course, whether or not you have actually "bought" anything.[2]

After our brief introduction to the World Wide Web in this chapter, and once we have dealt with the nitty-gritty of actually "getting onto the web" in the next chapter, we will start our discussion of the following web technologies, in the order listed:

- *HTML* will provide us with a way to describe the *structure* of each document we wish to place on our website.
- *CSS* will allow us to give each of our web documents the layout and presentational style we wish it to have.
- We will also use CSS to give a brief introduction to *responsive design*, which is the idea that a well-designed website should be able to recognize the kind of device, and the size of the viewport, upon which it is being displayed, and adjust itself accordingly.
- Web forms give our website visitors a chance to enter data that we can use in conducting business transactions with those visitors.
- *JavaScript* is a "client-side" programming language for performing computations and input data validation, as well as user interactivity, in the browser context. (See the next two items as well.)
- *DOM* (*Document Object Model*) provides a tree-like structural model of a web document, and an API (*Application Programming Interface*) that allows languages like JavaScript to access and modify parts of a web page during and after its display, often under user control.
- *DHTML* (stands for *Dynamic HTML*, which is what the combined usage of HTML, CSS, JavaScript, and the DOM is generally called) is another acronym you will often encounter in the context of producing interactivity for the browser user, but it is not really a separate technology.
- *PHP* (*PHP: Hypertext Preprocessor*, originally *Personal Home Page*) is a server-side programming language that permits the creation, on the server, by running a PHP program (or *script*), web pages whose content can vary each time that page is created.
- *AJAX* (*Asynchronous JavaScript And XML*) is not a separate technology, but rather a combination of several preexisting technologies that we can use to refresh only part of a web page, thus avoiding the need to refresh a complete page when only a small part of it needs to change.
- *MySQL* is the implementation of a relational database system, which can be used to store virtually any kind of business data, and that we can access and use on our websites.
- Web access to a MySQL database on a server can be performed by a PHP program on that server and controlled from our website.

[2] Software that is actually free should not be confused with shareware, which is software that (usually) you may use without restriction for a while, but you are required to pay for it eventually if you continue to use it. This requirement is often based on the honor system, and whether the developer gets paid is thus dependent on the integrity of the user.

- *XML* (*eXtensible Markup Language*) provides a text-based and very flexible way of describing data of any kind and, together with some associated technologies, permits such data to be easily transferred between all manner of applications, on and off the web.
- *DTD* (*Document Type Definition)* is a frequently used method for validation of XML documents.
- *XSLT* (*eXtensible Stylesheet Language Transformations*) is a language for transforming an XML file in various ways, including into HTML for display in a browser.
- Collection and analysis of website visitor data on the server via such techniques as web logs and cookies.

All of the technologies we will discuss are freely available, either automatically simply because you have installed a web browser, or by downloading the appropriate software from the Internet. There are many additional options we will not discuss that are also freely available. However, there is an equally large number of proprietary solutions available from companies like **Microsoft** (*Active Server Pages*, or *ASP*) and **Adobe** (*Flash*). Flash was formerly a Macromedia product, but Adobe has acquired Macromedia.

1.8 Some Alternative Technologies and Additional Tools

Some of the technologies mentioned previously that we will cover in later chapters are absolutely essential for any web developer to know. At a bare minimum, these include HTML, CSS, and JavaScript. In fact, on the client side of the browser-server relationship, each of these is basically "the only game in town" for its particular purpose. However, on the sever side, our choice of the combination of PHP and MySQL is only one of many different alternatives among competing technologies. These range from *Perl*, which is a much older, but very powerful and flexible, programming language that uses something called the *Common Gateway Interface* (*CGI*) for performing the same kinds of actions for which we will use PHP, all the way up to *Node.js*, which is a relatively recent server-side technology that has allowed JavaScript to become a powerful tool on the server as well as in the browser.

Keep in mind that this is a text about fundamentals, the nitty-gritty of web design. Learning what we have to show you here will stand you in good stead for years to come. Although there will certainly be changes and extensions to HTML, CSS, and JavaScript, the essentials will remain the same. And even if you choose not to use PHP and/or MySQL, what we cover when we discuss those technologies will still be useful if you choose one of the alternatives.

You will also likely have heard about many fabulous CSS and JavaScript libraries and frameworks, and even complete solutions to the problem of setting up a website and getting it online with almost no effort. In this context the names and acronyms you are likely to encounter include Bootstrap, Foundation, HTML5 Boilerplate, AngularJS, Backbone.js, Ember, Knockout, jQuery, Prototype, script.aculo.us, MooTools, Dojo, the Yahoo! UI Library, WordPress, Django, Joomla! . . . and the list goes on and on.

Each one of these tools, and many more like them that are available now and will appear in the future, may well have a place in the web developer's toolbox. Whether any one of them should become resident in your toolbox is a choice you should wait to make, until you have a reasonable grasp of the fundamentals and can properly evaluate a particular tool for whatever web development activity you plan to undertake. You should definitely be aware that these tools exist, and some of them are very, very useful and widely used. jQuery is a prime example of one in this category, for example. Do not think we are discouraging their use; it's simply that our focus is elsewhere. We have to walk before we can run, let alone soar . . .

Summary

In this chapter, we looked at some of the key concepts of interest to those approaching web development. These included the Internet, the World Wide Web itself, web browsers, and web servers. We also gained some insight into communication on the web and discussed some of the major protocols, such as TCP/IP and HTTP. We also learned about IP addresses, Fully Qualified Domain Names, and how DNS are used to convert one to the other. We also distinguished between URLs, URNs, and URIs. Finally, we got a glimpse of an actual e-commerce website. We closed the chapter by briefly describing the technologies that will be used to develop various features of our own website and that are commonly encountered in many e-commerce websites, including HTML, CSS, JavaScript, AJAX, PHP, and MySQL.

Quick Questions to Test Your Basic Knowledge

In this section of each chapter you will find questions that invite short answers, and you should be able to find those answers directly in, or at least be able to infer those answers directly from, the material in the current chapter.

1. What are three different ways computers can be "connected" to one another? The fact that "connected" is enclosed in quotes is a hint to one of the ways.
2. What is the difference between the Internet and the World Wide Web?
3. What does the acronym ARPANET stand for, and what is its relation to the Internet?
4. If each decade is divided into two "periods", an "early half" and a "late half" (early 1990s and late 1990s, for example), in what period was the Internet "invented", and in what period was the World Wide Web "invented"?
5. What would you give as a single sentence to describe the relationship between CERN, HTML, and Tim Berners-Lee?

6. What is the term that refers to the "linking together" of documents so that one can move easily from viewing one document to viewing another document to which it is linked?
7. What was the name of the first widely used text-based browser, and the name of the first widely used GUI-based browser?
8. What is the term used to describe the "architecture" illustrated by the relationship between a web browser and a web server?
9. What are the two (potentially confusing) meanings of the term *web server*?
10. What are the two web browsers that currently have the most users? Note that the answer to this question may well depend on when it is being asked?
11. What is a "communication protocol"?
12. What is the acronym for the underlying web protocol that is used nearly universally these days to move information across the Internet, and what does that acronym stand for?
13. When and why might you find it useful to use the UDP web protocol?
14. If a friend tells you she has just used ftp, what has she probably done?
15. What are the names of two different models that describe the "layered architecture" used when a program on one computer communicates over the Internet with a program on another computer?
16. What form does an IP address have?
17. Can you give an example of a valid IP address and an invalid IP address and state why the first is valid and the second is not?
18. What is IPv6, and what problem will it solve?
19. What is an FQDN, and how does it relate to an IP address?
20. What are the two possible meanings of the acronym DNS?
21. What are the names and brief descriptions of each part of the URL shown below?

```
http://cs.smu.ca/~pawan/opinions/comments.txt
```

22. What is the difference between a URL, a URN, and a URI?

Short Exercises to Improve Your Basic Understanding

In this section of each chapter you will find short, mostly hands-on, exercises that ask you to perform activities based on the material of the current chapter.

1. Find a computer connected to the Internet with at least two web browsers installed on it. Choose several different websites (the home pages of Microsoft, Google, Adobe, and YouTube, for example). Use each browser to visit each of those pages and note any differences between the appearance of each page from one browser to the next. This is a good way to convince yourself that the "web experience" is not yet as consistent as we might like it

to be. It is also a good way to convince you at the outset that we must always be vigilant in constructing our web pages to be as certain as we can be that we have minimized any differences that will be seen by visitors to our web pages who happen to be using different web browsers.

2. On the basis of the preceding exercise, formulate a *best practice* that you (and *all* web developers) should follow consistently.
3. Locate a real-world e-commerce website (other than the Jones & Bartlett Learning site used in this chapter) that actually sells products online. Browse the site as though you were going to be an actual customer, choosing items and placing them in the site's "shopping cart", perhaps deleting some items along the way, and then proceeding to check out before canceling the transaction. This will give you a sense of where we are heading in our discussions in the rest of this book.
4. In this chapter we made the statement that IPv6 would provide "more Internet addresses than we are ever likely to need". Do a search to find out how many addresses it will actually provide, and decide whether you agree with that statement.
5. The problem of running out of IP addresses is not quite as dire as we might have suggested, because of something called *Network Address Translation* (*NAT*, yet another acronym). This technology permits one unique address on the Internet to "map" into many other addresses that are "hidden" behind that one that appears on the Internet, thus expanding the effective number of available IP addresses. Do a search for more information on NAT if this idea intrigues you. In fact, you may already be using NAT in your home network.

Exercises on the Parallel Project

Since this is the first chapter, and the first group of exercises of this type, we need to say a few words about what we mean by our idea of a "parallel project".

Beginning in Chapter 2 we will be using the example of a business called **Nature's Source** that sells health products online to illustrate all of the topics for web development that we discuss. This example will extend throughout the text, and we want you to develop your own "parallel business" and its corresponding "parallel website" by implementing the same functionality for your business and website that you see in our text example.

Your task for this first parallel project exercise is to produce and submit a single textfile, as described in the specifications given in what follows.

So let's get started. The first thing you need to do is to think about what kind of "business" you would like to run, and for which you would like to develop a website. Our only restrictions[3] are that it cannot be an online store for health products, which is our own main example. It is

[3] The institutional environment in which you are working may have other restrictive criteria of which we cannot possibly be aware, but of which you should be aware. Decency, for example, or political correctness.

worth giving some thought to this choice, since you will be committed to it for the rest of your work with this text, unless you want to make major and time-consuming changes midstream. If you know of an actual existing business that does not have a website and would like one, that too is a possibility to consider.

The main thing to keep in mind is that for the time being you are to think mainly about website **content** related to your business, because you will be producing only a single textfile. In other words, there is no place for any styling or other enhancements at the moment. Nevertheless, it will be helpful to begin thinking about the **structure** of your content, because the next step (in the next "parallel project" exercise) will be to describe that structure using HTML element tags (i.e., to create a "real" web page). And that page and subsequent pages will be based on the text **content** you create for this exercise, as well as any content you add later.

This does not mean that you won't be able to make changes later. It's just that it's always a good idea to minimize the need for later revisions as much as possible. So, it will save you time later if you choose carefully now the wording of any paragraphs that you eventually want to go on your "home page", as well as on other "pages" of the website for your business. The advantage of making such decisions now is that you will have to worry about it less in later submissions when you are turning your plain text into HTML elements, and splitting your content into several logical parts to put into several separate files to create several new website pages. And remember . . . even though it's "only" text you are preparing, it should at least be formatted in such a way that the logical grouping of the various parts of the text makes sense to a reader.

Keep in mind as you work that you are in fact preparing web page content for a business, so do not put anything on in that you would not want a visitor to your site to see, or that does not make sense in that context. For example, there should be no indication that this is an assignment in a course! The bottom line here is that you must have the proper "website mind-set" as you prepare your content.

Since your website must be developed in such a way that it "parallels" the sample **Nature's Source** website developed in the text, it will be very helpful to look ahead at the various versions of the text website as they appear in the upcoming chapters. As you do this, note that all but one of the links on our home page are "generic" in the sense that they could apply to any online business. The one exception is the **Your Health** menu option, which is specific to the kind of business conducted by **Nature's Source**.

Having chosen the type of your business, you now need to "flesh it out" by making some additional choices, and here is a description of the (minimal amount of) content you must have in the textfile for this exercise:

1. Begin with a title that contains the name of your business.
2. Next, include several lines giving location and contact information for your business.
3. Follow this with two or three paragraphs providing some relevant information that serves to introduce your visitors to your business. This is information that will serve as the main

content of your home page as your website develops. It should be whatever you would want your visitors to see and read about your business when they first come to the business's website. Your mind-set must be that you do have a business, and you want to grab the attention of your website's visitors and get them interested in buying your products and/or employing your services.

4. Next comes a list of your products and/or services. These must appear in several categories (at least four) and each category must contain several products or services (at least six). The details will, of course, depend on the nature of your business, but if you cannot come up with at least this many offerings, perhaps you should consider going with another business. This is information that will eventually be moved to other pages on your website and, much later, into a database for your "e-store".
5. Add a paragraph or two giving a brief history of your company and how it developed (make this up if you don't have an actual business). Think of it as something that might eventually go under an **About Us** link.
6. Add a paragraph or two giving a brief "vision and mission statement" summarizing your company's approach to its business (again, make this up if you don't have an actual business). This too could be another sub-item of **About Us**.
7. Now choose an aspect of your business that is special with respect to your business (analogous to the **Your Health** category of **Nature's Source**), and add a couple of paragraphs about that part of your business.
8. Finally, add a copyright notice to establish "ownership" of your website and its content. This would be a good place to put your name and might perhaps be the only place where your name should appear.

Once you have made these choices, enter them into a textfile called my_business.txt. Make sure that the file content is well organized from a conceptual point of view, and pay particular attention to spelling and grammar. You can never assume that your viewers (and potential customers) will not be put off by bad spelling and poor grammar. Also, be prepared to submit this file for approval in some way that will be described by your instructor if you are required to do so.

What Else You May Want or Need to Know

This section of each chapter will contain items that may simply extend or enhance material in the chapter in some way, questions for which you may need to search elsewhere for the answers, or further exercises to help you consolidate and extend your knowledge and understanding of the chapter material and possibly how it relates to the material of one or more other chapters. A Google search may often be useful.

1. There are probably more web browsers than you thought there were. Try to name 10 different web browsers and the platform(s) on which they run (since some browsers are available on more than one platform).
2. During the early days of the web, many people used things that had names like Gopher, Veronica, Archie, and Jughead. What were these "technologies" and why are we no longer using them today?
3. What does the acronym W3C stand for?
4. What is the IETF and what does it do?
5. In the context of the Internet, what does the acronym RFC stand for, and what role does an RFC play in the ongoing development of the Internet?
6. The two HTTP *request methods* that are most often used in communication between web browsers and web servers are GET and POST. Describe briefly the main difference between them. We will come back to these much later in the text, but you may wish to explore them now for a sneak preview.
7. What are some of the other, less frequently used, HTTP request methods and for what purpose is each one used?
8. When an HTTP request is made by a browser to a server, a *status code* is returned by the server to the browser. Often the user does not see, and may not even be aware of, these status codes. But . . . what is (probably) the most commonly encountered HTTP status code that an average user will actually see?
9. Name the seven layers of the *Open Systems Interconnect Model*, as well as the four layers of the *Internet Model*, and indicate where the two models "match up".
10. We mentioned that the part of a URL to the right of the rightmost period in the URL is the "largest domain" in that URL and is generally either a two-letter *country code*, like ca for Canada or au for Australia, or a three-or-more-letter designation like gov for a government department, com for a commercial enterprise, or edu for an educational institution. Newer designations in this last category are gradually being accepted, such as mobi for mobile-compatible sites and travel for travel and tourist industry-related sites. Do a search to see if you can find an up-to-date list of Internet top-level domains.
11. The full syntax of a URL is actually somewhat more complicated than we showed in this chapter. In all its glory it can look something like this:

    ```
    scheme://username:password@domain:port/
    path?query_string#anchor
    ```

    Here the username:password will be required if the resource is password protected, though the more usual approach is to have the user fill in a *login form* that requires a username and corresponding password. The :port part of the URL may be required to indicate the *port number* on which the server is "listening" for requests to access a particular resource. By default, web servers generally listen on port number 80, and other standard services have their own assigned port numbers, so whether this part of the URL is required depends on the situation. The query_string that is separated from the path portion of the URL by a question mark (?) represents some information that is being sent to the server from the browser, and will likely be "processed" by the destination resource on the server. This information might be data from a web page form, for example. We will come back to this later in the text as well. And finally, the #anchor represents a special marker on a web page that tells the browser to go to that part of the page and start its display from there rather than from the beginning of the page. If you keep an eye on your browser's address window as you surf the web, you may see some or all of these features in the URLs that you observe.

12. As you browse the Internet you will of course be using the http protocol scheme (or its secure counterpart, https) as you visit various sites. But sometimes you wish to open, in your browser, a document or some other "resource" that is located on your own computer. In this case, the corresponding file is not being "served" by a web server; instead it is simply being opened in the browser in much the same way it would be opened by any other program running on your computer. However, there is another protocol scheme that the browser uses to deal with files in this situation, the file protocol. For example, if you have the file test.html on your Windows PC in the location

```
C:\MyWork\web\test.html
```

and you open this file in your browser and then check your browser's address window you will likely see this:

```
file:///C:/MyWork/web/test.html
```

Here, file:// corresponds to http:// in the sense that the general syntax of a URI requires that it start with the name of the protocol scheme, followed by a colon (:) and two forward slashes (//). The third forward slash may be interpreted as the "top-level directory" of your computer, and the rest is just the path down to the file. Note that even though Windows uses backslashes (\) to separate portions of a path, these have become forward slashes, which is the Linux/Unix way of doing things, and is also the generic format for paths in this context. You may also type a file URI directly into your browser address window if you know the local path to the file.

References

Most of our references are links to Internet sites, including many to Wikipedia. Keep in mind, though, that the web is a very dynamic place, and sites come and go, or change their URLs. Thus some of the links we provide may have changed or even disappeared when you try them. If this turns out to be the case, you can probably find the new location of the site or a different but comparable site by going to Google and providing a few appropriate search terms. Such is the wonderfully convenient nature of today's World Wide Web. As for Wikipedia itself, we have found it to be a reasonably reliable source of useful and up-to-date information. It is, of course, always a good policy, which we try to follow ourselves and which we recommend to you, to verify the accuracy of any information you get from the Internet or any other source, by comparing what several different sources say about it.

1. You can read about the history of the World Wide Web at this location:

   ```
   http://en.wikipedia.org/wiki/History_of_the_World_Wide_Web
   ```

2. "The World Wide Web Consortium (W3C) is an international community that develops standards to ensure the long-term growth of the web." This is a quote from their website, which you can find at:

   ```
   http://www.w3.org/
   ```

   From here you can follow links to read about all the web-related standards the Consortium has approved or has under development.
3. You can find a great deal of information that allows you to compare many different browsers in various ways at the following site:

   ```
   http://en.wikipedia.org/wiki/Comparison_of_web_browsers
   ```

4. The Web Standards Project has prepared some "Acid Tests" browser users can employ if they wish to see how well their browser of choice does in handling web standards. If you're interested, check out this link:

   ```
   http://www.acidtests.org/
   ```

5. The acronym *gTLD* stands for *generic Top Level Domain*. Why is it a small g? Who knows? But you can find a list of country codes for top-level domains that are countries, and a list of the other top-level domains as well, at this location:

   ```
   http://en.wikipedia.org/wiki/
   List_of_Internet_top-level_domains
   ```

   This should remain a relatively dynamic site, as newer top-level domains get added to the growing pool.
6. For further information on the URL/URN/URI question, check out the following site:

   ```
   http://en.wikipedia.org/wiki/Uniform_Resource_Locator
   ```

7. You can probably find more details on URL syntax than you will ever need to know at this site:

   `http://www.w3.org/Addressing/URL/5_BNF.html`

8. For more information on the file scheme for accessing files on your computer with your browser, see:

   `http://en.wikipedia.org/wiki/File_URI_scheme`

9. Here are URLs for some of the useful tools we mentioned, but will not discuss:

   `http://getbootstrap.com/`

   `http://foundation.zurb.com/`

   `https://html5boilerplate.com/`

   `https://angularjs.org/`

   `http://backbonejs.org/`

   `http://emberjs.com/`

   `http://knockoutjs.com/`

   `https://jquery.com/`

   `http://prototypejs.org/`

   `https://script.aculo.us/`

   `http://mootools.net/`

   `https://dojotoolkit.org/`

   `http://yuilibrary.com/`

   `https://wordpress.com/`

   `https://www.djangoproject.com/`

   `http://www.joomla.org/`