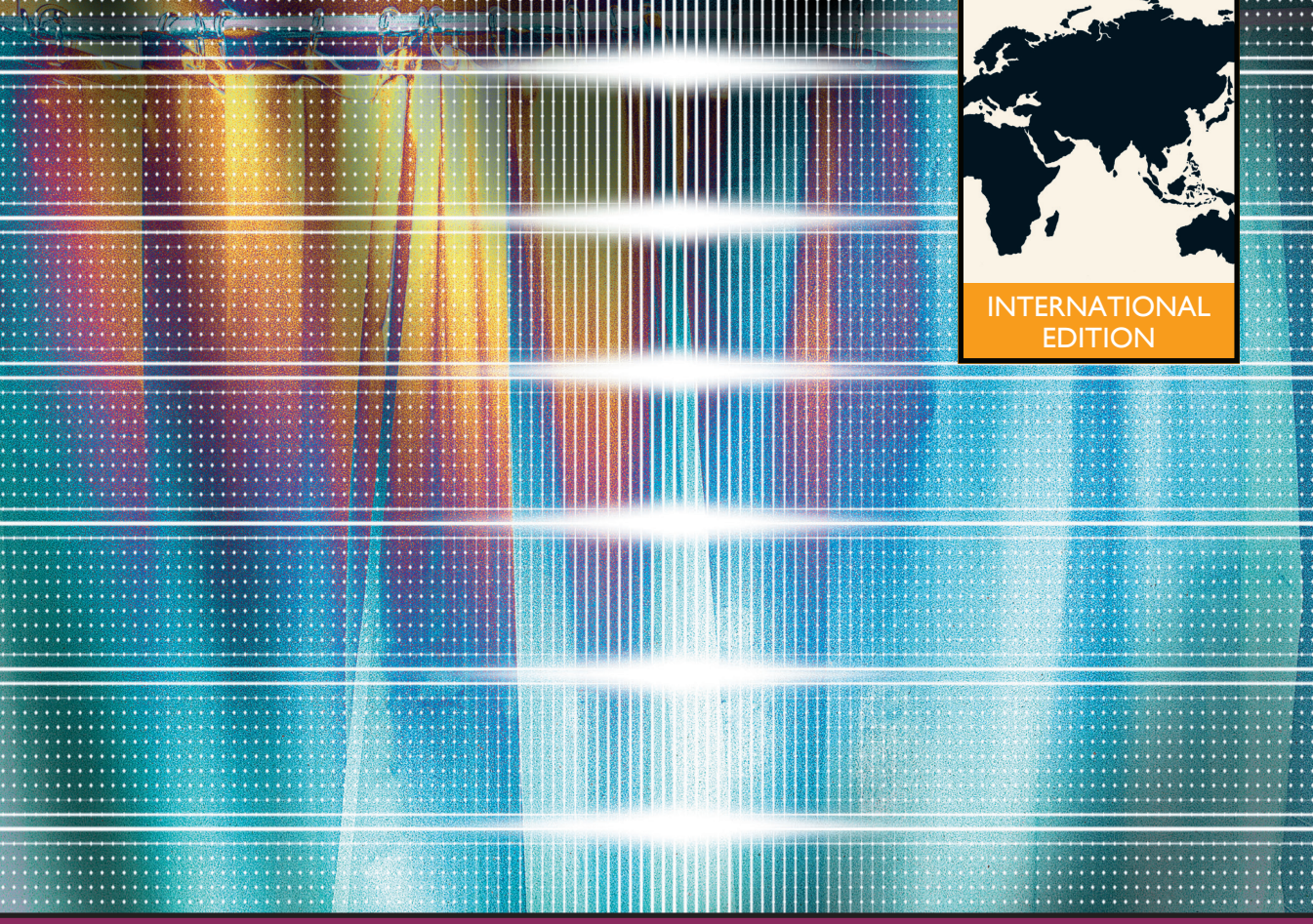


Internet & World Wide Web

HOW TO PROGRAM

Fifth Edition

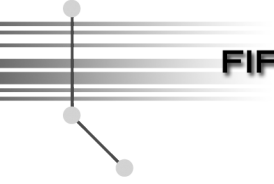
Paul Deitel • Harvey Deitel • Abbey Deitel

ALWAYS LEARNING

PEARSON

Internet & World Wide Web

HOW TO PROGRAM



FIFTH EDITION

Deitel Series Page

How To Program Series

C++ How to Program, 8/E
C How to Program, 6/E
Java™ How to Program, 9/E
Java™ How to Program, Late Objects Version, 8/E
Internet & World Wide Web How to Program, 5/E
Visual C++® 2008 How to Program, 2/E
Visual Basic® 2010 How to Program
Visual C#® 2010 How to Program, 3/E
Small Java™ How to Program, 6/E
Small C++ How to Program, 5/E

Simply Series

Simply C++: An App-Driven Tutorial Approach
Simply Java™ Programming: An App-Driven Tutorial Approach
Simply C#: An App-Driven Tutorial Approach
Simply Visual Basic® 2008, 3/E: An App-Driven Tutorial Approach

CourseSmart Web Books

www.deitel.com/books/CourseSmart/

C++ How to Program, 5/E, 6/E, 7/E & 8/E
Simply C++: An App-Driven Tutorial Approach
Java™ How to Program, 6/E, 7/E, 8/E & 9/E
Simply Visual Basic 2008: An App-Driven Tutorial Approach, 3/E

(continued next column)

(continued)

Visual Basic® 2010 How to Program
Visual Basic® 2008 How to Program
Visual C#® 2010 How to Program, 4/E
Visual C#® 2008 How to Program, 3/E

Deitel® Developer Series

AJAX, Rich Internet Applications and Web Development for Programmers
Android for Programmers: An App-Driven Approach
C++ for Programmers
C# 2010 for Programmers, 3/E
iPhone® for Programmers: An App-Driven Approach
Java™ for Programmers, 2/e
JavaScript for Programmers

LiveLessons Video Learning Products

www.deitel.com/books/LiveLessons/

Android App Development Fundamentals
C++ Fundamentals
Java™ Fundamentals
C# Fundamentals
iPhone® App Development Fundamentals
JavaScript Fundamentals
Visual Basic Fundamentals

To receive updates on Deitel publications, Resource Centers, training courses, partner offers and more, please register for the free *Deitel Buzz Online* e-mail newsletter at:

www.deitel.com/newsletter/subscribe.html

and join the Deitel communities on Twitter®

@deitel

and Facebook®

facebook.com/DeitelFan/

To communicate with the authors, send e-mail to:

deitel@deitel.com

For information on government and corporate *Dive-Into Series* on-site seminars offered by Deitel & Associates, Inc. worldwide, visit:

www.deitel.com/training/

or write to

deitel@deitel.com

For continuing updates on Prentice Hall/Deitel publications visit:

www.deitel.com
www.pearsoninternationaleditions.com/deitel/

Visit the Deitel Resource Centers that will help you master programming languages, software development, Android and iPhone/iPad app development, and Internet- and web-related topics:

www.deitel.com/ResourceCenters.html

Internet & World Wide Web

HOW TO PROGRAM

FIFTH EDITION

Paul Deitel
Deitel & Associates, Inc.

Harvey Deitel
Deitel & Associates, Inc.

Abbey Deitel
Deitel & Associates, Inc.

International Edition contributions by

Soumen Mukherjee ✦ Arup Kumar Bhattacharjee

PEARSON

Boston Columbus Indianpolis New York San Francisco Upper Saddle River
Amsterdam Cape Town Dubai London Madrid Milan Munich Paris Montreal Toronto
Delhi Mexico City Sao Paulo Sydney Hong Kong Seoul Singapore Taipei Tokyo

Editorial Director: ***Marcia J. Horton***
Editor-in-Chief: ***Michael Hirsch***
Associate Editor: ***Carole Snyder***
Vice President, Marketing: ***Patrice Jones***
Marketing Manager: ***Yezan Alayan***
Marketing Coordinator: ***Kathryn Ferranti***
Vice President, Production: ***Vince O'Brien***
Managing Editor: ***Jeff Holcomb***
Associate Managing Editor: ***Robert Engelhardt***
Publisher, International Edition: ***Angshuman Chakraborty***
Acquisitions Editor, International Edition: ***Somnath Basu***
Publishing Assistant, International Edition: ***Shokhi Shah***
Print and Media Editor, International Edition: ***Ashwitha Jayakumar***
Project Editor, International Edition: ***Jayashree Arunachalam***
Operations Specialist: ***Lisa McDowell***
Art Director: ***Anthony Gemmellaro***
Media Editor: ***Daniel Sandin***

Pearson Education Limited
Edinburgh Gate
Harlow
Essex CM20 2JE
England

and Associated Companies throughout the world

Visit us on the World Wide Web at:
www.pearsoninternationaleditions.com

Authorized adaptation from the United States edition, entitled Internet & World Wide Web, How to Program, 5th edition, *ISBN 978-0-13-215100-9by Paul Deitel, Harvey Deitel and Abbey Deitel published by Pearson Education © 2012.*

ISBN 10: 0-273-76402-0
ISBN 13: 978-0-273-76402-1

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library

10 9 8 7 6 5 4 3 2 1
14 13 12 11 10

Typeset in AGaramond-Regular by GEX Publishing Services
Printed and bound by Courier, Westford in The United States of America

The publisher's policy is to use paper manufactured from sustainable forests.

In memory of Paul Baran,
designer of a survivable distributed communications network and packet switching, which are the basis for the protocols used on the Internet today.

Paul, Harvey and Abbey Deitel

Trademarks

Apache is a trademark of The Apache Software Foundation.

Apple, iPhone, iPad, iOS and Safari are registered trademarks of Apple, Inc.

CSS, DOM, XHTML and XML are trademarks of the World Wide Web Consortium.

Firefox is a registered trademark of the Mozilla Foundation.

Google is a trademark of Google, Inc.

JavaScript, Java and all Java-based marks are trademarks or registered trademarks of Oracle in the United States and other countries.

Microsoft, Internet Explorer, Silverlight and the Windows logo are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Opera is a trademark of Opera Software.

Contents

Chapters 24–29 and Appendices E–F are PDF documents posted online at the book's Companion Website (located at www.pearsoninternationaleditions.com/deitel/).

Preface

Science and technology and the various forms of art,
all unite humanity in a single and interconnected system.

—Zhores Aleksandrovich Medvede

Welcome to Internet and web programming with *Internet & World Wide Web How to Program, Fifth Edition*! This book presents leading-edge computing technologies for students, instructors and software developers.

The world of computing—and Internet and web programming in particular—has changed dramatically since the last edition. This new edition focuses on HTML5 and the related technologies in its ecosystem, diving into the exciting new features of HTML5, CSS3, the latest edition of JavaScript (ECMAScript 5) and HTML5 `canvas`. We focus on popular key technologies that will help you build Internet- and web-based applications that interact with other applications and with databases. These form the basis of the kinds of enterprise-level, networked applications that are popular in industry today.

Internet & World Wide Web How to Program, 5/e is appropriate for both introductory and intermediate-level client-side and server-side programming courses. The book is also suitable for professionals who want to update their skills with the latest Internet and web programming technologies.

At the heart of the book is the Deitel signature "live-code approach"—concepts are presented in the context of complete working HTML5 documents, CSS3 stylesheets, JavaScript scripts, XML documents, programs and database files, rather than in code snippets. Each complete code example is accompanied by live sample executions. The source code is available at `www.deitel.com/books/iw3htp5/` and at the book's Companion Website `www.pearsoninternationaleditions.com/deitel/`.

As you read the book, if you have questions, send an e-mail to `deitel@deitel.com`; we'll respond promptly. For updates on this book, visit `www.deitel.com/books/iw3htp5/`, join our communities on Facebook (`www.facebook.com/deitelfan`) and Twitter (`@deitel`), and subscribe to the *Deitel® Buzz Online* newsletter (`www.deitel.com/newsletter/subscribe.html`).

New and Updated Features

Here are the updates we've made for *Internet & World Wide Web How to Program, 5/e*:

- ***New Chapter 1.*** The new Chapter 1 engages students with intriguing facts and figures to get them excited about studying Internet and web applications development. The chapter includes a table of some of the research made possible by

computers and the Internet, current technology trends and hardware discussion, the data hierarchy, a new section on social networking, a table of popular web services, a table of business and technology publications and websites that will help you stay up to date with the latest technology news and trends, and updated exercises.

- ***New HTML5 features.*** Chapter 3 introduces the latest features of HTML5 including the new HTML5 form input types and page structure elements (Fig. 1). *The new HTML5 features are not universally implemented in all of the web browsers.* This is changing as the browser vendors release new versions. We discuss many additional HTML5 features throughout the book.

New HTML5 features			
Form Input Types			
`color`	`date`	`datetime`	`datetime-local`
`email`	`month`	`number`	`range`
`search`	`tel`	`time`	`url`
`week`	`input` element	`datalist` element	`autocomplete` attribute
Page Structure Elements			
`header`	`nav`	`figure`	`figcaption`
`article`	`summary`	`section`	`aside`
`meter`	`footer`	text-level semantics (marking potential line breaks)	

Fig. 1 | New HTML5 form input types and page structure elements

- ***New CSS3 features.*** Chapter 5 introduces the latest features of CSS3 (Fig. 2). *The new CSS3 features are not universally implemented in all of the web browsers.* This is changing as the browser vendors release new versions.

New CSS3 features		
text shadows	rounded corners	color
box shadows	linear gradients	radial gradients
multiple background images	image borders	animations
transitions	transformations	`@font-face` rule
Flexible Box Layout Module	`:nth-child` selectors	multicolumn layouts
media queries		
Non-standard features		
text stroke	reflection	

Fig. 2 | New CSS3 features.

- ***Updated treatment of JavaScript.*** We've strengthened the JavaScript coverage in Chapters 6–16. JavaScript has become the *de facto* standard client-side scripting language for web-based applications due to its highly portable nature. Our treatment, which is appropriate for novices, serves two purposes—it introduces client-side scripting (Chapters 6–16), which makes web pages more dynamic and interactive, and it provides the programming foundation for the server-side scripting in PHP presented in Chapter 19. JavaScript looks similar to basic core language features in C, C++, C# and Java. Once you learn JavaScript, you've got a foothold on learning these other popular programming languages.
- ***New HTML5 canvas.*** Chapter 14 replaces the Flash and Silverlight chapters from the previous edition with the new HTML5 `canvas` element for 2D graphics (Fig. 3). `canvas` is built into the browser, eliminating the need for plug-ins like Flash and Silverlight, and helping you improve performance and convenience, and reduce costs. At the end of the chapter, you'll use `canvas` to build a fun, animated Cannon Game with audio effects, which we built in Flash in previous editions of this book.

HTML5 canvas features		
rectangles	lines	arcs and circles
shadows	quadratic curves	Bezier curves
linear gradients	radial gradients	image manipulation
images	patterns	transformations
alpha transparency	compositing	

Fig. 3 | HTML5 `canvas` features.

- ***New and updated multimedia exercises.*** Chapter 14 includes several new and updated multimedia exercises (Fig. 4).

New and updated multimedia exercises		
Cannon Game Enhancements	Random Interimage Transition	Digital Clock
Animation	Scrolling Image Marquee	Background Audio
Scrolling Marquee Sign	Automatic Jigsaw Puzzle Generator	Analog Clock
Dynamic Audio and Graphical Kaleidoscope	Horse Race	Maze Generator and Walker
One-Armed Bandit	Fireworks Designer	Shuffleboard
Game of Pool	15 Puzzle	Floor Planner
Crossword Puzzle	Coloring Black-and-White Photographs and Images	Reaction Time Tester
Rotating Images		Vacuuming Robot
		Eyesight Tester

Fig. 4 | New and updated multimedia exercises.

- ***Tested on seven browsers.*** For the last edition of this book, we tested all the code on two desktop browsers—Internet Explorer and Firefox. For this new edition, we tested all of the code in the most current versions of *seven* popular browsers—five for the **desktop** (**Chrome**, **Internet Explorer**, **Firefox**, **Opera** and **Safari**) and two for **mobile** devices (**iPhone/iPad** and **Android**). *HTML5 and CSS3 are evolving and the final standards have not been approved yet.* The browser vendors are selectively implementing features that are likely to be standardized. Some vendors have higher levels of feature compliance than others. With each new version of the browsers, the trend has been to significantly increase the amount of functionality that's been implemented. The HTML5 test site (`html5test.com`) measures how well each browser supports the pending standards and specifications. You can view test scores and see which features are supported by each browser. You can also check sites such as `http://caniuse.com/` for a list of features covered by each browser. *Not every document in this book will render properly in each browser.* Instead of choosing only capabilities that exist universally, we demonstrate exciting new features in whatever browser handles the new functionality best. As you read this book, run each example in multiple web browsers so you can view and interact with it as it was originally intended. And remember, things are changing quickly, so a browser that did not support a feature when we wrote the book could support it when you read the book.
- ***Validated HTML5, CSS3 and JavaScript code.*** All of the HTML5, CSS3 and JavaScript code in the book was validated using **`validator.w3.org/`** for HTML5, **`jigsaw.w3.org/css-validator`** for CSS3 and **`javascriptlint.com`** for JavaScript. *Not every script fully validates but most do.* Although all of the code works properly, you may receive warnings (or possibly errors) when validating code with some of the new features.
- ***Smartphone and tablet apps.*** You're probably familiar with the explosion of apps available for the **iPhone/iPad** and **Android** platforms. There's almost a million apps between the two. Previously, writing apps for these platforms required detailed knowledge of each, and in the case of iPhone/iPad, was strictly controlled by Apple; Android is more open. With the techniques you'll learn in this book, you'll be able to write apps that are portable between a great variety of desktop and mobile platforms, including iPhone/iPad and Android. You'll even be able to sell those apps on your own terms (or through certain app stores as well). This is an exciting possibility! It's one of the true virtues of developing with HTML5, CSS3 and JavaScript in general, and HTML5 `canvas` in particular. Running an HTML5 app on your smartphone or tablet is as simple as opening it in your compliant web browser. *You may still encounter some portability issues.*
- ***New HTML5 web storage capabilities.*** In Chapter 11, we use HTML5's new web storage capabilities to create a web application that stores a user's favorite Twitter searches on the computer for easy access at a later time. Web storage replaces the controversial cookie technology, offering lots more storage space. Chapter 11 also briefly introduces JSON, a means for creating JavaScript objects—typically for transferring data over the Internet between client-side and server-side programs.

- ***Enhanced Craps game featuring HTML5 audio and video elements.*** The Craps game in Chapter 9 now includes an HTML5 `audio` element that plays a dice-rolling sound each time the user rolls the dice. Also, we link to a page with an embedded HTML5 `video` element that plays a video explaining the rules of the game.
- ***jQuery Ajax case study.*** The previous edition of this book included a calendar application that used the Dojo libraries—which were popular at the time—to create the user interface, communicate with the server asynchronously, handle events and manipulate the DOM. Since then, jQuery has become the most popular JavaScript library. For this edition, we've updated the calendar application (Chapter 16) using jQuery and placed it online as a jQuery Ajax case study.
- ***New HTML5 WebSockets and Web Workers capabilities.*** We've added an online treatment of two new technologies—WebSockets, which provides a simple model for networking, and Web Workers which provides multithreading on a web page.
- ***Ajax-enabled web applications.*** We've updated the chapter on building Ajax-enabled web applications, with applications that demonstrate *partial-page updates* and type-ahead capabilities—each of these are key capabilities of Rich Internet Applications.
- ***HTML DOM and XML DOM.*** We've enhanced the treatments of HTML DOM manipulation, JavaScript events and XML DOM manipulation with JavaScript.
- ***LINQ.*** Since the last edition of the book, Microsoft introduced LINQ (Language-Integrated Query) to replace SQL for database access. Chapter 18 provides an introduction to LINQ basics and an introduction to LINQ to SQL (the technology that replaces SQL).
- ***Updated PHP coverage.*** Chapter 19 has been updated to the latest version of PHP. If you start this book as a novice and study the JavaScript in Chapters 6–13, you'll have the programming experience needed to understand server-side programming in PHP. [Our treatment of server-side programming in ASP.NET requires knowledge of C# or Visual Basic, and in JSF requires knowledge of Java.]
- ***ASP.NET, ASP.NET Ajax and web services.*** This updated three-chapter sequence is now provided for each of Microsoft's two key applications development languages—C# and Visual Basic. The C# chapters and the first VB chapter are in the print book and the remaining Visual Basic chapters are available online at the book's Companion Website (see the inside front cover).
- ***JavaServer Faces (JSF), JSF Ajax and web services.*** This updated three-chapter sequence, available online, emphasizes building Ajax-enabled JSF applications.
- ***Web services.*** We now provide chapters on building both SOAP-based web services *and* REST-based web services with ASP.NET in Visual Basic, ASP.NET in C# and JSF in Java.
- ***Client/Server applications.*** Several client-side case studies now enable students to interact with preimplemented web services that we host at `test.deitel.com`.

- *New and updated case studies.* The book includes rich case studies using various technologies—Deitel Cover Viewer (JavaScript/DOM), Address Book (Ajax), Cannon Game (HTML5 Canvas), Mailing List (PHP/MySQL), Guest Book and Password-Protected Books Database (ASP.NET), Address Book (JavaServer Faces) and Blackjack (JAX-WS web services).

New Pedagogic Features

- *Making a Difference exercises in Chapter 1.* We encourage you to use computers and the Internet to research and solve significant social problems. These exercises are meant to increase awareness and discussion of important issues the world is facing. We hope you'll approach them with your own values, politics and beliefs. Check out the many Making a Difference resources we provide, including our new Making a Difference Resource Center at `www.deitel.com/MakingADifference` for additional ideas you may want to investigate further.
- *Page numbers for key terms in chapter summaries.* For key terms that appear in the Chapters 1–19 summaries, we include the page number of the key term's defining occurrence in the text.

Dependency Chart

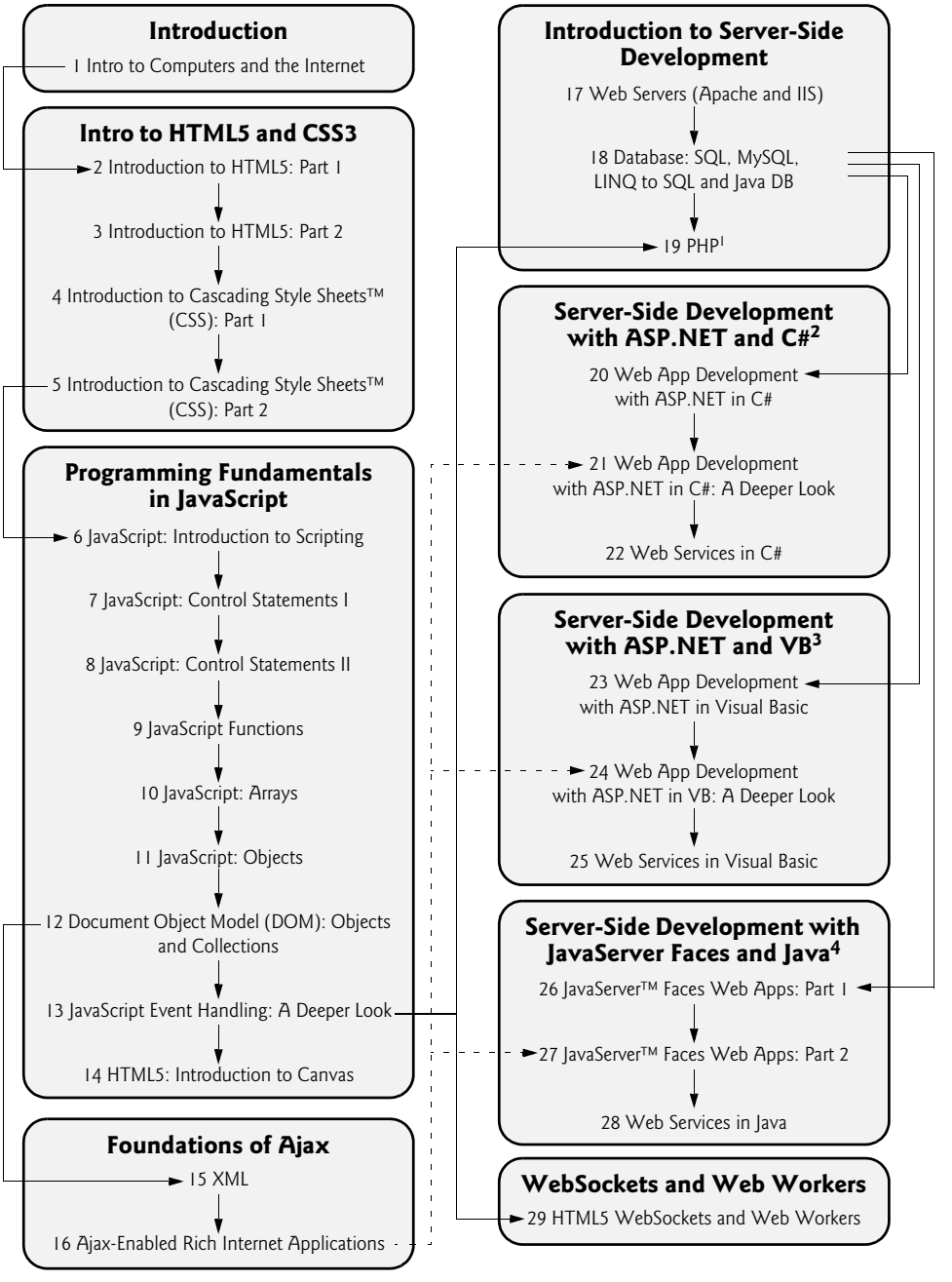
The chart in Fig. 5 shows the book's modular organization and the dependencies among the chapters to help instructors plan their syllabi. *Internet & World Wide Web How to Program, 5/e*, is appropriate for a variety of introductory and intermediate -level programming courses, most notably client-side programming and server-side programming. Chapters 1–23 are in the printed book; Chapters 24–29 and some appendices are online.

We recommend that you study all of a given chapter's dependencies before studying that chapter, though other orders are certainly possible. Some of the dependencies apply only to sections of chapters, so we advise instructors to browse the material before designing a course of study. This book is intended for courses that teach pure client-side web programming, courses that teach pure server-side web programming, and courses that mix and match some of each. Readers interested in studying server-side technologies should understand how to build web pages using HTML5 and CSS3, and object-based programming in JavaScript. Chapters 15 and 16 can be taught as part of a client-side unit, at the beginning of a server-side unit or split between the two.

HTML5 Accessibility Online Appendix

According to the W3C Web Accessibility Initiative, your web pages and applications should be accessible so that "people with disabilities can perceive, understand, navigate, and interact with the web, and that they can contribute to the web."[1] In an online appendix, we enumerate accessibility issues you should consider when designing web pages and web-based applications. We also provide resources that show you how to use HTML5, CSS3, JavaScript and various design techniques to create accessible web pages and applications. As appropriate, we tie the information in this appendix back to the appropriate chapters and sections so that you can see how the applications may be enhanced to improve web accessibility.

1. `http://www.w3.org/WAI/intro/accessibility.php`.



1. Chapter 19 assumes only that you're familiar with the programming fundamentals presented in Chapters 6–13.
2. The C# chapters require knowledge of C# and the Microsoft .NET class libraries.
3. The Visual Basic chapters require knowledge of Visual Basic and the Microsoft .NET class libraries.
4. The Java chapters require knowledge of Java and the Java class libraries.

Fig. 5 | *Internet & World Wide Web How to Program, 5/e* chapter dependency chart.

HTML5 Geolocation Online Appendix

The HTML5 Geolocation API allows you to build web applications that gather location information (i.e,. latitude and longitude coordinates) using technologies like GPS, IP addresses, WiFi connections or cellular tower connections. It's supported by the seven desktop and mobile browsers we used to test the code throughout the book.

The Geolocation API specification lists several use cases,[2] including:

- finding points of interest in the user's area
- annotating content with location information
- showing the user's position on a map
- providing route navigation
- alerting the user when points of interest are nearby
- providing up-to-date local information
- tagging locations in status updates on social networking sites

For example, you could create a location-based mobile web app that uses GPS location information from a smartphone to track a runner's route on a map, calculate the distance traveled and the average speed. Similarly, you could create an app that returns a list of nearby businesses. In this online appendix, we build a mobile location-based app.

Teaching Approach

Internet & World Wide Web How to Program, 5/e, contains hundreds of complete working examples across a wide variety of markup, styling, scripting and programming languages. We stress clarity and concentrate on building well-engineered software.

Syntax Shading. For readability, we syntax shade the code, similar to the way most integrated-development environments and code editors syntax color the code. Our syntax-shading conventions are:

```
comments appear like this
keywords appear like this
constants and literal values appear like this
all other code appears in black
```

Code Highlighting. We place gray rectangles around key code segments.

Using Fonts for Emphasis. We place the key terms and the index's page reference for each defining occurrence in **bold** text for easy reference. We emphasize on-screen components in the **bold Helvetica** font (for example, the **File** menu) and program text in the `Lucida` font (for example, `int count = 5`).

Web Access. All of the source-code examples can be downloaded from:

```
www.deitel.com/books/iw3htp5
www.pearsoninternationaleditions.com/deitel
```

Objectives. The opening quotes are followed by a list of chapter objectives.

2. `http://www.w3.org/TR/geolocation-API/#usecases_section`.

Illustrations/Figures. Abundant tables, line drawings, documents, scripts, programs and program outputs are included.

Programming Tips. We include programming tips to help you focus on important aspects of software development. These tips and practices represent the best we've gleaned from a combined seven decades of programming and teaching experience.

Good Programming Practices

The Good Programming Practices *call attention to techniques that will help you produce programs that are clearer, more understandable and more maintainable.*

Common Programming Errors

Pointing out these Common Programming Errors *reduces the likelihood that you'll make them.*

Error-Prevention Tips

These tips contain suggestions for exposing and removing bugs from your programs; many of the tips describe aspects of programming that prevent bugs from getting into programs.

Performance Tips

These tips highlight opportunities for making your scripts and programs run faster or minimizing the amount of memory that they occupy.

Portability Tips

The Portability Tips *help you write code that will run on a variety of platforms.*

Software Engineering Observations

The Software Engineering Observations *highlight architectural and design issues that affect the construction of software systems, especially large-scale systems.*

Summary Bullets. We present a section-by-section bullet-list summary of the chapter for rapid review of key points. For ease of reference, we include the page number of each key term's defining occurrence in the text.

Self-Review Exercises and Answers. Extensive self-review exercises *and* answers are included for self study.

Exercises. The chapter exercises include:

- simple recall of important terminology and concepts
- What's wrong with this code?
- writing individual statements
- writing complete functions and scripts
- major projects

Index. We've included an extensive index. Defining occurrences of key terms are highlighted with a **bold** page number.

Instructor Resources

The following supplements are available to *qualified instructors only* through Pearson Education's Instructor Resource Center (www.pearsoninternationaleditions.com/deitel):

- ***PowerPoint® slides*** containing all the code and figures in the text, plus bulleted items that summarize key points.
- ***Solutions Manual*** with solutions to many of the end-of-chapter exercises. Please check the Instructor Resource Center to determine which exercises have solutions.

Please do not write to us requesting access to the Pearson Instructor's Resource Center. Access is restricted to college instructors teaching from the book. Instructors may obtain access only through their Pearson representatives. If you're not a registered faculty member, contact your Pearson representative.

Solutions are *not* provided for "project" exercises. Check out our Programming Projects Resource Center for lots of additional exercise and project possibilities (www.deitel.com/ProgrammingProjects/).

Acknowledgments

We'd like to thank Barbara Deitel for long hours devoted to this project. We're fortunate to have worked with the dedicated team of publishing professionals at Pearson. We appreciate the guidance, savvy and energy of Michael Hirsch, Editor-in-Chief of Computer Science. Carole Snyder recruited the book's reviewers and managed the review process. Bob Engelhardt managed the book's production.

Reviewers

We wish to acknowledge the efforts of our fourth and fifth edition reviewers. They scrutinized the text and the programs and provided countless suggestions for improving the presentation: Timothy Boronczyk (Consultant), Roland Bouman (MySQL AB), Chris Bowen (Microsoft), Peter Brandano (KoolConnect Technologies, Inc.), Matt Chotin (Adobe), Chris Cornutt (PHPDeveloper.org), Phil Costa (Adobe), Umachitra Damodaran (Sun Microsystems), Vadiraj Deshpande (Sun Microsystems), Justin Erenkrantz (The Apache Software Foundation), Christopher Finke (Netscape), Jesse James Garrett (Adaptive Path), Mike Harsh (Microsoft), Chris Heilmann (Mozilla), Kevin Henrikson (Zimbra.com), Tim Heuer (Microsoft), Molly E. Holtzschlag (W3C), Ralph Hooper (University of Alabama, Tuscaloosa), Chris Horton (University of Alabama), John Hrvatin (Microsoft), Johnvey Hwang (Splunk, Inc.), Joe Kromer (New Perspective and the Pittsburgh Adobe Flash Users Group), Jennifer Kyrnin (Web Design Guide at About.com), Eric Lawrence (Microsoft), Pete LePage (Microsoft), Dr. Roy Levow (Florida Atlantic University), Billy B. L. Lim (Illinois State University), Shobana Mahadevan (Sun Microsystems), Patrick Mineault (Freelance Flash Programmer), Anand Narayanaswamy (Microsoft), John Peterson (Insync and V.I.O., Inc.), Jennifer Powers (University of Albany), Ignacio Ricci (Ignacioricci.com), Jake Rutter (onerutter.com), Robin Schumacher (MySQL AB), José Antonio González Seco (Parlamento de Andalucia), Dr. George Semeczko (Royal & SunAlliance Insurance Canada), Steven Shaffer (Penn State University), Michael Smith (W3C), Karen Tegtmeyer (Model Technologies, Inc.), Paul Vencill (MITRE), Raymond Wen (Microsoft), Eric M. Wendelin (Auto-trol Technology

Corporation), Raymond F. Wisman (Indiana University), Keith Wood (Hyro, Ltd.) and Daniel Zappala (Brigham Young University).

As you read the book, we'd appreciate your comments, criticisms, corrections and suggestions for improvement. Please address all correspondence to:

```
deitel@deitel.com
```

We'll respond promptly. We hope you enjoy working with *Internet & World Wide Web How to Program, 5/e*.

Paul, Harvey and Abbey Deitel

About the Authors

Paul J. Deitel, CEO and Chief Technical Officer of Deitel & Associates, Inc., is a graduate of MIT, where he studied Information Technology. Through Deitel & Associates, Inc., he has delivered hundreds of Java, C++, C, C#, Visual Basic and Internet programming courses to industry clients, including Cisco, IBM, Siemens, Sun Microsystems, Dell, Lucent Technologies, Fidelity, NASA at the Kennedy Space Center, the National Severe Storm Laboratory, White Sands Missile Range, Rogue Wave Software, Boeing, SunGard Higher Education, Stratus, Cambridge Technology Partners, One Wave, Hyperion Software, Adra Systems, Entergy, CableData Systems, Nortel Networks, Puma, iRobot, Invensys and many more. He and his co-author, Dr. Harvey M. Deitel, are the world's best-selling programming-language textbook authors.

Dr. Harvey M. Deitel, Chairman and Chief Strategy Officer of Deitel & Associates, Inc., has 50 years of experience in the computer field. Dr. Deitel earned B.S. and M.S. degrees from MIT and a Ph.D. from Boston University. He has extensive college teaching experience, including earning tenure and serving as the Chairman of the Computer Science Department at Boston College before founding Deitel & Associates, Inc., with his son, Paul J. Deitel. He and Paul are the co-authors of dozens of books and LiveLessons video packages and they are writing many more. The Deitels' texts have earned international recognition, with translations published in Japanese, German, Russian, Chinese, Spanish, Korean, French, Polish, Italian, Portuguese, Greek, Urdu and Turkish. Dr. Deitel has delivered hundreds of professional programming seminars to major corporations, academic institutions, government organizations and the military.

Abbey Deitel, President of Deitel & Associates, Inc., is a graduate of Carnegie Mellon University's Tepper School of Management where she received a B.S. in Industrial Management. Abbey has been managing the business operations of Deitel & Associates, Inc. for 14 years. She has contributed to numerous Deitel & Associates publications and is the co-author of *iPhone for Programmers: An App-Driven Approach* and *Android for Programmers: An App-Driven Approach*.

Corporate Training from Deitel & Associates, Inc.

Deitel & Associates, Inc., is an internationally recognized corporate training and authoring organization. The company provides instructor-led courses delivered at client sites worldwide on major programming languages and platforms, such as Java™, C++, Visual C++®, C, Visual C#®, Visual Basic®, XML®, Python®, object technology, Internet and web programming,

Android™ and iPhone® app development, and a growing list of additional programming and software-development courses. The founders of Deitel & Associates, Inc., are Paul J. Deitel and Dr. Harvey M. Deitel. The company's clients include many of the world's largest companies, government agencies, branches of the military, and academic institutions. Through its 36-year publishing partnership with Prentice Hall/Pearson, Deitel & Associates publishes leading-edge programming textbooks, professional books and *LiveLessons* video courses. Deitel & Associates, Inc., and the authors can be reached via e-mail at:

```
deitel@deitel.com
```

To learn more about the company, its publications and its *Dive Into® Series* Corporate Training curriculum delivered at client locations worldwide, visit:

```
www.deitel.com/training/
```

subscribe to the *Deitel® Buzz Online* e-mail newsletter at:

```
www.deitel.com/newsletter/subscribe.html
```

and join the authors' communities on Facebook (`www.facebook.com/DeitelFan`) and Twitter (`@deitel`).

Individuals wishing to purchase Deitel books, and *LiveLessons* video training courses can do so through `www.deitel.com`. Bulk orders by corporations, the government, the military and academic institutions should be placed directly with Pearson. For more information, visit

```
www.pearsoninternationaleditions.com/deitel
```

Before You Begin

Please follow these instructions to download the book's examples and ensure you have a current web browser before you begin using this book.

Obtaining the Source Code

The examples for *Internet & World Wide Web How To Program, 5/e* are available for download at

```
www.deitel.com/books/iw3htp5/
```

If you're not already registered at our website, go to `www.deitel.com` and click the **Register** link below our logo in the upper-left corner of the page. Fill in your information. There's no charge to register, and we do not share your information with anyone. We send you only account-management e-mails unless you register separately for our free *Deitel® Buzz Online* e-mail newsletter at `www.deitel.com/newsletter/subscribe.html`. After registering for the site, you'll receive a confirmation e-mail with your verification code. *Click the link in the confirmation e-mail to complete your registration*. Configure your e-mail client to allow e-mails from `deitel.com` to ensure that the confirmation email is not filtered as junk mail.

Next, go to `www.deitel.com` and sign in using the **Login** link below our logo in the upper-left corner of the page. Go to `www.deitel.com/books/iw3htp5/`. You'll find the link to download the examples under the heading **Download Code Examples and Other Premium Content for Registered Users**. Write down the location where you choose to save the ZIP file on your computer. Extract the example files to your hard disk using a ZIP file extractor program. If you are working in a computer lab, ask your instructor where you can save the example code.

Web Browsers Used in This Book

We tested all of the code in the most current versions of *seven* popular browsers—five for the **desktop** (**Chrome**, **Internet Explorer**, **Firefox**, **Opera** and **Safari**) and two for **mobile** devices (**iPhone** and **Android**). HTML5 and CSS3 are evolving and the final standards have not been approved yet. The browser vendors are selectively implementing features that are likely to become a part of the standards. Some vendors have higher levels of feature compliance than others. With each new version of the browsers, the trend has been to significantly increase the amount of functionality that's been implemented. The HTML5 test site (`html5test.com`) measures how well each browser supports the pending standards and specifications. You can view test scores and see which features are supported by each browser. You can also check sites such as `http://caniuse.com/` for a list of features covered by each browser. Not every document in this book will render properly in each browser. Instead of choosing only capabilities that exist universally, we demonstrate exciting new

features in whatever browser handles the new functionality best. As you read this book, run each example in multiple web browsers so you can view and interact with it as it was originally intended. And remember, things are changing quickly, so a browser that did not support a feature when we wrote the book could support it when you read the book.

Web Browser Download Links

You can download the desktop browsers from the following locations:

- Google Chrome: `http://www.google.com/chrome`
- Mozilla Firefox: `http://www.mozilla.org/firefox/new/`
- Microsoft Internet Explorer (Windows only): `http://www.microsoft.com/ie`
- Apple Safari: `http://www.apple.com/safari/`
- Opera: `http://www.opera.com/`

We recommend that you install all the browsers that are available for your platform.

Software for the C# and Visual Basic ASP.NET Chapters

The C# (Chapters 20–22) and Visual Basic (Chapters 23–25) ASP.NET and web services chapters require Visual Web Developer 2010 Express and SQL Server 2008 Express. These tools are downloadable from `www.microsoft.com/express`. You should follow the default installation instructions for each.

Software for the JavaServer Faces and Java Web Services Chapters

The software required for the JavaServer Faces and Java Web Services chapters (Chapters 26–28) is discussed at the beginning of Chapter 26.

You're now ready to begin your web programming studies with *Internet & World Wide Web How to Program, 5/e*. We hope you enjoy the book! If you have any questions, please feel free to email us at `deitel@deitel.com`. We'll respond promptly.

Introduction to Computers and the Internet

1

People are using the web to build things they have not built or written or drawn or communicated anywhere else.
—Tim Berners-Lee

How wonderful it is that nobody need wait a single moment before starting to improve the world.
—Anne Frank

Man is still the most extraordinary computer of all.
—John F. Kennedy

Objectives

In this chapter you'll learn:

- Computer hardware, software and Internet basics.
- The evolution of the Internet and the World Wide Web.
- How HTML5, CSS3 and JavaScript are improving web-application development.
- The data hierarchy.
- The different types of programming languages.
- Object-technology concepts.
- And you'll see demos of interesting and fun Internet applications you can build with the technologies you'll learn in this book.

Outline

1.1 Introduction

Welcome to the exciting and rapidly evolving world of Internet and web programming! There are more than two billion Internet users worldwide—that's approximately 30% of the Earth's population.[1] In use today are more than a billion general-purpose computers, and billions more *embedded* computers are used in cell phones, smartphones, tablet computers, home appliances, automobiles and more—and many of these devices are connected to the Internet. According to a study by Cisco Internet Business Solutions Group, there were 12.5 billion Internet-enabled devices in 2010, and the number is predicted to reach 25 billion by 2015 and 50 billion by 2020.[2] The Internet and web programming technologies you'll learn in this book are designed to be *portable*, allowing you to design web pages and applications that run across an enormous range of Internet-enabled devices.

You'll begin by learning the *client-side programming* technologies used to build web pages and applications that are run on the *client* (i.e., in the browser on the user's device). You'll use HyperText Markup Language 5 (HTML5) and Cascading Style Sheets 3 (CSS3)—the recent releases of HTML and CSS technologies—to add powerful, dynamic and fun features and effects to web pages and web applications, such as audio, video, animation, drawing, image manipulation, designing pages for multiple screen sizes, access to web storage and more.

You'll learn *JavaScript*—the language of choice for implementing the client side of Internet-based applications (we discuss JavaScript in more detail in Section 1.3). Chapters 6–13 present rich coverage of JavaScript and its capabilities. You'll also learn about *jQuery*—the JavaScript library that's dramatically reshaping the world of web development. Throughout the book there's also an emphasis on *Ajax* development, which helps you create better-performing, more usable applications.

Later in the book, you'll learn *server-side programming*—the applications that respond to requests from client-side web browsers, such as searching the Internet, checking your

1. www.internetworldstats.com/stats.htm.
2. www.cisco.com/web/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf.

bank-account balance, ordering a book from Amazon, bidding on an eBay auction and ordering concert tickets. We present condensed treatments of four popular Internet/web programming languages for building the server side of Internet- and web-based client/server applications. Chapters 19–22 and 23–28 present three popular server-side technologies, including PHP, ASP.NET (in both C# and Visual Basic) and JavaServer Faces.

Be sure to read both the Preface and the Before You Begin section to learn about the book's coverage and how to set up your computer to run the hundreds of code examples. The code is available at `www.deitel.com/books/iw3htp5` and `www.pearsonhighered.com/deitel`. Use the source code we provide to *run every program and script* as you study it. Try each example in *multiple browsers*. If you're interested in smartphones and tablet computers, be sure to run the examples in your browsers on iPhones, iPads, Android smartphones and tablets, and others. The technologies covered in this book and browser support for them are evolving rapidly. *Not every feature of every page we build will render properly in every browser*. All seven of the browsers we use are free.

Moore's Law

Every year, you probably expect to pay at least a little more for most products and services. The opposite has been the case in the computer and communications fields, especially with regard to the costs of hardware supporting these technologies. For many decades, hardware costs have fallen rapidly. Every year or two, the capacities of computers have approximately *doubled* inexpensively. This remarkable trend often is called **Moore's Law**, named for the person who identified it, Gordon Moore, co-founder of Intel—the leading manufacturer of the processors in today's computers and embedded systems. Moore's Law and related observations apply especially to the amount of memory that computers have for programs, the amount of secondary storage (such as disk storage) they have to hold programs and data over longer periods of time, and their processor speeds—the speeds at which computers execute their programs (i.e., do their work). Similar growth has occurred in the communications field, in which costs have plummeted as enormous demand for communications bandwidth (i.e., information-carrying capacity) has attracted intense competition. We know of no other fields in which technology improves so quickly and costs fall so rapidly. Such phenomenal improvement is truly fostering the *Information Revolution*.

1.2 The Internet in Industry and Research

These are exciting times in the computer field. Many of the most influential and successful businesses of the last two decades are technology companies, including Apple, IBM, Hewlett Packard, Dell, Intel, Motorola, Cisco, Microsoft, Google, Amazon, Facebook, Twitter, Groupon, Foursquare, Yahoo!, eBay and many more. These companies are major employers of people who study computer science, information systems or related disciplines. At the time of this writing, Apple was the most valuable company in the world.

In the past, most computer applications ran on computers that were not connected to one another, whereas today's Internet applications can be written to communicate among computers throughout the world.

Figures 1.1–1.4 provide a few examples of how computers and the Internet are being used in industry and research. Figure 1.1 lists two examples of how computers and the Internet are being used to improve health care.

Name	Description
Electronic health records	These might include a patient's medical history, prescriptions, immunizations, lab results, allergies, insurance information and more. Making this information available to health care providers across a secure network improves patient care, reduces the probability of error and increases overall efficiency of the health care system.
Human Genome Project	The Human Genome Project was founded to identify and analyze the 20,000+ genes in human DNA. The project used computer programs to analyze complex genetic data, determine the sequences of the billions of chemical base pairs that make up human DNA and store the information in databases which have been made available over the Internet to researchers in many fields.

Fig. 1.1 | Computers and the Internet in health care.

Figure 1.2 provides a sample of some of the exciting ways in which computers and the Internet are being used for social good. In the exercises at the end of this chapter, you'll be asked to propose other projects that would use computers and the Internet to "make a difference."

Name	Description
AMBER™ Alert	The AMBER (America's Missing: Broadcast Emergency Response) Alert System is used to find abducted children. Law enforcement notifies TV and radio broadcasters and state transportation officials, who then broadcast alerts on TV, radio, computerized highway signs, the Internet and wireless devices. AMBER Alert recently partnered with Facebook, whose users can "Like" AMBER Alert pages by location to receive alerts in their news feeds.
World Community Grid	People worldwide can donate their unused computer processing power by installing a free secure software program that allows the World Community Grid (`www.worldcommunitygrid.org`) to harness unused capacity. This computing power, accessed over the Internet, is used in place of expensive supercomputers to conduct scientific research projects that are making a difference, providing clean water to third-world countries, fighting cancer, growing more nutritious rice for regions fighting hunger and more.
One Laptop Per Child (OLPC)	One Laptop Per Child (`one.laptop.org`) is providing low-power, inexpensive, Internet-enabled laptops to poor children worldwide—enabling learning and reducing the digital divide.

Fig. 1.2 | Projects that use computers and the Internet for social good.

We rely on computers and the Internet to communicate, navigate, collaborate and more. Figure 1.3 gives some examples of how computers and the Internet provide the infrastructure for these tasks.

Name	Description
Cloud computing	**Cloud computing** allows you to use software, hardware and information stored in the "cloud"—i.e., accessed on remote computers via the Internet and available on demand—rather than having it stored on your personal computer. Amazon is one of the leading providers of public cloud computing services. You can rent extra storage capacity using the Amazon Simple Storage Service (Amazon S3), or augment processing capabilities with Amazon's EC2 (Amazon Elastic Compute Cloud). These services, allowing you to increase or decrease resources to meet your needs at any given time, are generally more cost effective than purchasing expensive hardware to ensure that you have enough storage and processing power to meet your needs at their peak levels. Business applications (such as CRM software) are often expensive, require significant hardware to run them and knowledgeable support staff to ensure that they're running properly and securely. Using cloud computing services shifts the burden of managing these applications from the business to the service provider, saving businesses money.
GPS	Global Positioning System (GPS) devices use a network of satellites to retrieve location-based information. Multiple satellites send time-stamped signals to the GPS device, which calculates the distance to each satellite based on the time the signal left the satellite and the time the signal arrived. This information is used to determine the exact location of the device. GPS devices can provide step-by-step directions and help you easily find nearby businesses (restaurants, gas stations, etc.) and points of interest. GPS is used in numerous location-based Internet services such as check-in apps to help you find your friends (e.g., Foursquare and Facebook), exercise apps such as RunKeeper that track the time, distance and average speed of your outdoor jog, dating apps that help you find a match nearby and apps that dynamically update changing traffic conditions.
Robots	Robots can be used for day-to-day tasks (e.g., iRobot's Roomba vacuum), entertainment (e.g., robotic pets), military combat, deep sea and space exploration (e.g., NASA's Mars rover) and more. RoboEarth (`www.roboearth.org`) is "a World Wide Web for robots." It allows robots to learn from each other by sharing information and thus improving their abilities to perform tasks, navigate, recognize objects and more.
E-mail, Instant Messaging, Video Chat and FTP	Internet-based servers support all of your online messaging. E-mail messages go through a mail server that also stores the messages. Instant messaging (IM) and Video Chat apps, such as AIM, Skype, Yahoo! Messenger and others allow you to communicate with others in real time by sending your messages and live video through servers. FTP (file transfer protocol) allows you to exchange files between multiple computers (e.g., a client computer such as your desktop and a file server) over the Internet using the TCP/IP protocols for transferring data.

Fig. 1.3 | Examples of computers and the Internet in infrastructure.

Figure 1.4 lists a few of the exciting ways in which computers and the Internet are used in entertainment.

Name	Description
iTunes and the App Store	iTunes is Apple's media store where you can buy and download digital music, movies, television shows, e-books, ringtones and apps (for iPhone, iPod and iPad) over the Internet. Apple's iCloud service allows you to store your media purchases "in the cloud" and access them from any iOS (Apple's mobile operating system) device. In June 2011, Apple announced at their World Wide Developer Conference (WWDC) that 15 billion songs had been downloaded through iTunes, making Apple the leading music retailer. As of July 2011, 15 billion apps had been downloaded from the App Store (`www.apple.com/pr/library/2011/07/07Apples-App-Store-Downloads-Top-15-Billion.html`).
Internet TV	Internet TV set-top boxes (such as Apple TV and Google TV) allow you to access an enormous amount of content on demand, such as games, news, movies, television shows and more.
Game programming	Global video game revenues are expected to reach $65 billion in 2011 (`uk.reuters.com/article/2011/06/06/us-videogames-factbox-idUKTRE75552I20110606`). The most sophisticated games can cost as much as $100 million to develop. Activision's *Call of Duty 2: Modern Warfare*, released in 2009, earned $310 million in just one day in North America and the U.K. (`news.cnet.com/8301-13772_3-10396593-52.html?tag=mncol;txt`)! Online *social gaming*, which enables users worldwide to compete with one another over the Internet, is growing rapidly. Zynga—creator of popular online games such as *Farmville* and *Mafia Wars*—was founded in 2007 and already has over 265 million monthly users. To accommodate the growth in traffic, Zynga is adding nearly 1,000 servers each week (`techcrunch.com/2010/09/22/zynga-moves-1-petabyte-of-data-daily-adds-1000-servers-a-week/`)!

Fig. 1.4 | Examples of computers and the Internet in entertainment.

1.3 HTML5, CSS3, JavaScript, Canvas and jQuery

You'll be learning the latest versions of several key client-side, web-application development technologies in this book. This section provides a brief overview of each.

HTML5

Chapters 2–3 introduce HTML (HyperText Markup Language)—a special type of computer language called a *markup language* designed to specify the *content* and *structure* of web pages (also called documents) in a portable manner. HTML5, now under development, is the emerging version of HTML. HTML enables you to create content that will render appropriately across the extraordinary range of devices connected to the Internet—including smartphones, tablet computers, notebook computers, desktop computers, special-purpose devices such as large-screen displays at concert arenas and sports stadiums, and more.

You'll learn the basics of HTML5, then cover more sophisticated techniques such as creating tables, creating forms for collecting user input and using new features in HTML5, including page-structure elements that enable you to give meaning to the parts of a page (e.g., headers, navigation areas, footers, sections, figures, figure captions and more).

A "stricter" version of HTML called *XHTML (Extensible HyperText Markup Language)*, which is based on XML (eXtensible Markup Language, introduced in Chapter 15), is still used frequently today. Many of the server-side technologies we cover later in the book produce web pages as XHTML documents, by default, but the trend is clearly to HTML5.

Cascading Style Sheets (CSS)

Although HTML5 provides some capabilities for controlling a document's presentation, *it's better not to mix presentation with content*. HTML5 should be used only to specify a document's structure and content.

Chapters 4–5 use **Cascading Style Sheets** (**CSS**) to specify the *presentation*, or styling, of elements on a web page (e.g., fonts, spacing, sizes, colors, positioning). CSS was designed to style portable web pages *independently* of their content and structure. By separating page styling from page content and structure, you can easily change the look and feel of the pages on an *entire* website, or a portion of a website, simply by swapping out one style sheet for another. CSS3 is the current version of CSS under development. Chapter 5 introduces many new features in CSS3.

JavaScript

JavaScript is a language that helps you build *dynamic* web pages (i.e., pages that can be modified "on the fly" in response to *events*, such as user input, time changes and more) and computer applications. It enables you to do the client-side programming of web applications. In addition, there are now several projects dedicated to *server-side* JavaScript, including CommonJS (`www.commonjs.org`), Node.js (`nodejs.org`) and Jaxer (`jaxer.org`).

JavaScript was created by Netscape, the company that built the first wildly successful web browser. Both Netscape and Microsoft have been instrumental in the standardization of JavaScript by ECMA International (formerly the European Computer Manufacturers Association) as ECMAScript. ECMAScript 5, the latest version of the standard, corresponds to the version of JavaScript we use in this book.

The JavaScript chapters of the book are more than just an introduction to the language. They also present computer-programming fundamentals, including control structures, functions, arrays, recursion, strings and objects. You'll see that JavaScript is a portable scripting language and that programs written in JavaScript can run in web browsers across a wide range of devices.

Web Browsers and Web-Browser Portability

Ensuring a consistent look and feel on client-side browsers is one of the great challenges of developing web-based applications. Currently, a standard does not exist to which software vendors must adhere when creating web browsers. Although browsers share a common set of features, each browser might render pages differently. Browsers are available in many versions and on many different platforms (Microsoft Windows, Apple Macintosh, Linux, UNIX, etc.). Vendors add features to each new version that sometimes result in cross-platform incompatibility issues. It's difficult to develop web pages that render correctly on all versions of each browser.

All of the code examples in the book were tested in the five most popular desktop browsers and the two most popular mobile browsers (Fig. 1.5). Support for HTML5, CSS3 and JavaScript features varies by browser. The *HTML5 Test* website (`http://html5test.com/`) scores each browser based on its support for the latest features of these

evolving standards. Figure 1.5 lists the five desktop browsers we use in reverse order of their HTML5 Test scores from most compliant to least compliant at the time of this writing. Internet Explorer 10 (IE10) is expected to have a much higher compliance rating than IE9. You can also check sites such as `http://caniuse.com/` for a list of features covered by each browser.

Portability Tip 1.1

The web is populated with many different browsers, including many older, less-capable versions, which makes it difficult for authors and web-application developers to create universal solutions. The W3C is working toward the goal of a universal client-side platform (`http://www.w3.org/2006/webapi/admin/charter`).

Browser	Approximate market share as of August 2011 (http://gs.statcounter.com)	Score out of 450 from html5test.com
Desktop browsers	***Market share***	
Google Chrome 13	17%	330
Mozilla Firefox 6	27%	298
Apple Safari 5.1	7%	293
Opera 11.5	2%	286
Internet Explorer 9	40%	141
Mobile browsers	***Mobile market share***	
iPhone	15% (of mobile browsers)	217
Android	18% (of mobile browsers)	184

Fig. 1.5 | HTML5 Test scores for the browsers used to test the examples.

jQuery

jQuery (`jQuery.org`) is currently the most popular of hundreds of *JavaScript libraries*.[3] jQuery simplifies JavaScript programming by making it easier to manipulate a web page's elements and interact with servers in a portable manner across various web browsers. It provides a library of custom graphical user interface (GUI) controls (beyond the basic GUI controls provided by HTML5) that can be used to enhance the look and feel of your web pages.

Validating Your HTML5, CSS3 and JavaScript Code

As you'll see, JavaScript programs typically have HTML5 and CSS3 portions as well. You must use proper HTML5, CSS3 and JavaScript syntax to ensure that browsers process your documents properly. Figure 1.6 lists the validators we used to validate the code in this book. Where possible, we eliminated validation errors.

3. `www.activoinc.com/blog/2008/11/03/jquery-emerges-as-most-popular-javascript-library-for-web-development/`.

Technology	Validator URL
HTML5	`http://validator.w3.org/` `http://html5.validator.nu/`
CSS3	`http://jigsaw.w3.org/css-validator/`
JavaScript	`http://www.javascriptlint.com/` `http://www.jslint.com/`

Fig. 1.6 | HTML5, CSS3 and JavaScript validators.

1.4 Demos

Browse the web pages in Fig. 1.7 to get a sense of some of the things you'll be able to create using the technologies you'll learn in this book, including HTML5, CSS3, JavaScript, canvas and jQuery. Many of these sites provide links to the corresponding source code, or you can view the page's source code in your browser.

URL	Description
`https://developer.mozilla.org/en-US/demos/`	Mozilla's DemoStudio contains numerous HTML5, canvas, CSS3 and JavaScript demos that use audio, video, animation and more.
`http://js-fireworks.appspot.com/`	Enter your name or message, and this JavaScript animation then writes it using a fireworks effect over the London skyline.
`http://9elements.com/io/projects/html5/canvas/`	Uses HTML5 canvas and audio elements to create interesting effects, and ties in tweets that include the words "HTML5" and "love" (click anywhere on the screen to see the next tweet).
`http://www.zachstronaut.com/lab/text-shadow-box/text-shadow-box.html`	Animated demo of the CSS3 text-shadow effect. Use the mouse to shine a light on the text and dynamically change the direction and size of the shadow.
`http://clublime.com/lab/html5/sphere/`	Uses an HTML5 canvas to create a sphere that rotates and changes direction as you move the mouse cursor.
`http://spielzeugz.de/html5/liquid-particles.html`	The Liquid Particles demo uses an HTML5 canvas. Move the mouse around the screen and the "particles" (dots or letters) follow.
`http://www.paulbrunt.co.uk/bert/`	Bert's Breakdown is a fun video game built using an HTML5 canvas.
`http://www.openrise.com/lab/FlowerPower/`	Canvas app that allows you to draw flowers on the page, adjust their colors, change the shapes of the petals and more.

Fig. 1.7 | HTML5, CSS3, JavaScript, canvas and jQuery demos. (Part 1 of 2.)

URL	Description
http://alteredqualia.com/canvasmol/	Uses canvas to display a 3D molecule that can be viewed from any desired angle (0–360 degrees).
http://pasjans-online.pl/	The game of Solitaire built using HTML5.
http://andrew-hoyer.com/experiments/cloth/	Uses canvas to simulate of the movement of a piece of cloth. Click and drag the mouse to move the fabric.
http://www.paulrhayes.com/experiments/cube-3d/	CSS3 demo allows you to use the mouse to tilt and rotate the 3D cube. Includes a tutorial.
http://www.effectgames.com/demos/canvascycle/	Animated waterfall provides a nice demo of using color in HTML5 canvas.
http://macek.github.com/google_pacman/	The Google PAC-MAN® game (a Google Doodle) built in HTML5.
http://www.benjoffe.com/code/games/torus/	A 3D game similar to Tetris® built with JavaScript and canvas.
http://code.almeros.com/code-examples/water-effect-canvas/	Uses canvas and JavaScript to create a water rippling effect. Hover the cursor over the canvas to see the effect. The site includes a tutorial.
http://jqueryui.com/demos/	Numerous jQuery demos, including animations, transitions, color, interactions and more.
http://lab.smashup.it/flip/	Demonstrates a flip box using jQuery.
http://tutorialzine.com/2010/09/html5-canvas-slideshow-jquery/	Slideshow built with HTML5 canvas and jQuery (includes a tutorial).
http://css-tricks.com/examples/Circulate/	Learn how to create an animated circulation effect using jQuery.
http://demo.tutorialzine.com/2010/02/photo-shoot-css-jquery/demo.html	Uses jQuery and CSS to create a photoshoot effect, allowing you to focus on an area of the page and snap a picture (includes a tutorial).

Fig. 1.7 | HTML5, CSS3, JavaScript, canvas and jQuery demos. (Part 2 of 2.)

1.5 Evolution of the Internet and World Wide Web

The Internet—a global network of computers—was made possible by the *convergence of computing and communications technologies*. In the late 1960s, ARPA (the Advanced Research Projects Agency) rolled out blueprints for networking the main computer systems of about a dozen ARPA-funded universities and research institutions. They were to be connected with communications lines operating at a then-stunning 56 Kbps (i.e., 56,000 bits per second)—this at a time when most people (of the few who could) were connecting over telephone lines to computers at a rate of 110 bits per second. A **bit** (short for "binary digit") is the smallest data item in a computer; it can assume the value 0 or 1.

There was great excitement. Researchers at Harvard talked about communicating with the powerful Univac computer at the University of Utah to handle the intensive cal-

culations related to their computer graphics research. Many other intriguing possibilities were raised. Academic research was about to take a giant leap forward. ARPA proceeded to implement the **ARPANET**, which eventually evolved into today's **Internet**.

Things worked out differently from what was originally planned. Rather than enabling researchers to share each other's computers, it rapidly became clear that communicating quickly and easily via electronic mail was the key early benefit of the ARPANET. This is true even today on the Internet, which facilitates communications of all kinds among the world's Internet users.

Packet Switching

One of the primary goals for ARPANET was to allow *multiple* users to send and receive information simultaneously over the *same* communications paths (e.g., phone lines). The network operated with a technique called **packet switching**, in which digital data was sent in small bundles called **packets**. The packets contained *address*, *error-control* and *sequencing* information. The address information allowed packets to be *routed* to their destinations. The sequencing information helped in reassembling the packets—which, because of complex routing mechanisms, could actually arrive out of order—into their original order for presentation to the recipient. Packets from different senders were intermixed on the same lines to efficiently use the available bandwidth. This packet-switching technique greatly reduced transmission costs, as compared with the cost of dedicated communications lines.

The network was designed to operate without centralized control. If a portion of the network failed, the remaining working portions would still route packets from senders to receivers over alternative paths for reliability.

TCP/IP

The protocol (i.e., set of rules) for communicating over the ARPANET became known as **TCP**—the **Transmission Control Protocol**. TCP ensured that messages were properly routed from sender to receiver and that they arrived intact.

As the Internet evolved, organizations worldwide were implementing their own networks for both intraorganization (i.e., within the organization) and interorganization (i.e., between organizations) communications. A wide variety of networking hardware and software appeared. One challenge was to get these different networks to communicate. ARPA accomplished this with the development of **IP**—the **Internet Protocol**, truly creating a **network of networks,** the current architecture of the Internet. The combined set of protocols is now commonly called **TCP/IP**. Each computer on the Internet has a unique **IP address**. The current IP standard, Internet Protocol version 4 (IPv4), has been in use since 1984 and will soon run out of possible addresses. The next-generation Internet Protocol, **IPv6**, is just starting to be deployed. It features enhanced security and a new addressing scheme, hugely expanding the number of IP addresses available so that we will not run out of IP addresses in the forseeable future.

Explosive Growth

Initially, Internet use was limited to universities and research institutions; then the military began using it intensively. Eventually, the government decided to allow access to the Internet for commercial purposes. The research and military communities were concerned that response times would become poor as the Internet became saturated with users.

In fact, the opposite has occurred. Businesses realized that they could tune their operations and offer new and better services to their clients, so they started spending vast amounts of money to develop and enhance the Internet. This generated fierce competition among communications carriers and hardware and software suppliers to meet this demand. The result is that **bandwidth** (i.e., the information-carrying capacity) on the Internet's is increasing rapidly as costs dramatically decline.

World Wide Web, HTML, HTTP

The **World Wide Web** allows computer users to execute web-based applications and to locate and view multimedia-based documents on almost any subject over the Internet. The web is a relatively recent creation. In 1989, **Tim Berners-Lee** of CERN (the European Organization for Nuclear Research) began to develop a technology for sharing information via hyperlinked text documents. Berners-Lee called his invention the **HyperText Markup Language** (**HTML**). He also wrote communication protocols to form the backbone of his new information system, which he called the World Wide Web. In particular, he wrote the **Hypertext Transfer Protocol** (**HTTP**)—a communications protocol used to send information over the web. The **URL** (**Uniform Resource Locator**) specifies the address (i.e., location) of the web page displayed in the browser window. Each web page on the Internet is associated with a unique URL. URLs usually begin with `http://`.

HTTPS

URLs of websites that handle private information, such as credit card numbers, often begin with `https://`, the abbreviation for **Hypertext Transfer Protocol Secure** (**HTTPS**). HTTPS is the standard for transferring encrypted data on the web. It combines HTTP with the Secure Sockets Layer (SSL) and the more recent Transport Layer Security (TLS) cryptographic schemes for securing communications and identification information over the web. Although there are many benefits to using HTTPS, there are a few drawbacks, most notably some performance issues because encryption and decryption consume significant computer processing resources.

Mosaic, Netscape, Emergence of Web 2.0

Web use exploded with the availability in 1993 of the Mosaic browser, which featured a user-friendly graphical interface. Marc Andreessen, whose team at the National Center for Supercomputing Applications (NCSA) developed Mosaic, went on to found Netscape, the company that many people credit with igniting the explosive Internet economy of the late 1990s. But the "dot com" economic bust brought hard times in the early 2000s. The resurgence that began in 2004 or so has been named **Web 2.0**. Google is widely regarded as the signature company of Web 2.0. Some other companies with Web 2.0 characteristics are YouTube (video sharing), Facebook (social networking), Twitter (microblogging), Groupon (social commerce), Foursquare (mobile check-in), Salesforce (business software offered as online services "in the cloud"), Craigslist (mostly free classified listings), Flickr (photo sharing), Skype (Internet telephony and video calling and conferencing, now owned by Microsoft) and Wikipedia (a free online encyclopedia).

1.6 Web Basics

In this section, we discuss the fundamentals of web-based interactions between a client web browser and a web server. In its simplest form, a *web page* is nothing more than an

HTML (HyperText Markup Language) document (with the extension `.html` or `.htm`) that describes to a web browser the document's content and structure.

Hyperlinks

HTML documents normally contain **hyperlinks**, which, when clicked, load a specified web document. Both images and text may be hyperlinked. When the mouse pointer hovers over a hyperlink, the default arrow pointer changes into a hand with the index finger pointing upward. Often hyperlinked text appears underlined and in a different color from regular text in a web page.

Originally employed as a publishing tool for scientific research, hyperlinks are widely used to reference sources, or sites that have more information on a particular topic. The paths created by hyperlinking create the effect of the "web."

When the user clicks a hyperlink, a **web server** locates the requested web page and sends it to the user's web browser. Similarly, the user can type the *address of a web page* into the browser's *address field* and press *Enter* to view the specified page.

Hyperlinks can reference other web pages, e-mail addresses, files and more. If a hyperlink's URL is in the form `mailto:`*emailAddress*, clicking the link loads your default e-mail program and opens a **message window** addressed to the specified e-mail address. If a hyperlink references a file that the browser is incapable of displaying, the browser prepares to **download** the file, and generally prompts the user for information about how the file should be stored. When a file is downloaded, it's copied onto the user's computer. Programs, documents, images, sound and video files are all examples of downloadable files.

URIs and URLs

URIs (Uniform Resource Identifiers) identify resources on the Internet. URIs that start with `http://` are called *URLs (Uniform Resource Locators)*. Common URLs refer to files, directories or server-side code that performs tasks such as database lookups, Internet searches and business-application processing. If you know the URL of a publicly available resource anywhere on the web, you can enter that URL into a web browser's address field and the browser can access that resource.

Parts of a URL

A URL contains information that directs a browser to the resource that the user wishes to access. Web servers make such resources available to web clients. Popular web servers include Apache's HTTP Server and Microsoft's Internet Information Services (IIS).

Let's examine the components of the URL

```
http://www.deitel.com/books/downloads.html
```

The text `http://` indicates that the HyperText Transfer Protocol (HTTP) should be used to obtain the resource. Next in the URL is the server's fully qualified **hostname** (for example, `www.deitel.com`)—the name of the web-server computer on which the resource resides. This computer is referred to as the **host**, because it houses and maintains resources. The hostname `www.deitel.com` is translated into an **IP (Internet Protocol) address**—a numerical value that uniquely identifies the server on the Internet. An Internet **Domain Name System (DNS) server** maintains a database of hostnames and their corresponding IP addresses and performs the translations automatically.

The remainder of the URL (`/books/downloads.html`) specifies the resource's location (`/books`) and name (`downloads.html`) on the web server. The location could represent an actual directory on the web server's file system. For *security* reasons, however, the location is typically a *virtual directory*. The web server translates the virtual directory into a real location on the server, thus hiding the resource's true location.

Making a Request and Receiving a Response

When given a web page URL, a web browser uses HTTP to request the web page found at that address. Figure 1.8 shows a web browser sending a request to a web server.

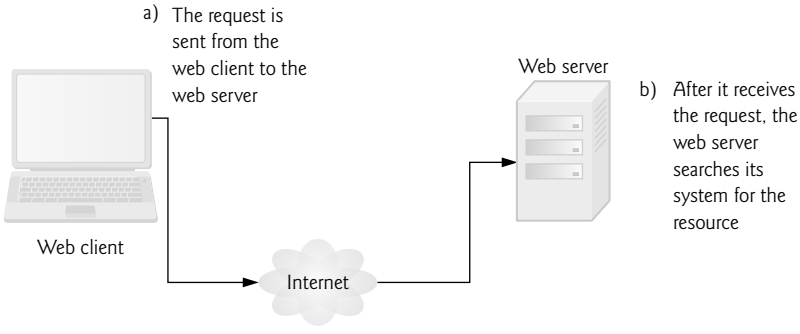


Fig. 1.8 | Client requesting a resource from a web server.

In Fig. 1.8, the web browser sends an HTTP request to the server. The request (in its simplest form) is

```
GET /books/downloads.html HTTP/1.1
```

The word **`GET`** is an **HTTP method** indicating that the client wishes to obtain a resource from the server. The remainder of the request provides the path name of the resource (e.g., an HTML5 document) and the protocol's name and version number (`HTTP/1.1`). The client's request also contains some required and optional headers.

Any server that understands HTTP (version 1.1) can translate this request and respond appropriately. Figure 1.9 shows the web server responding to a request.

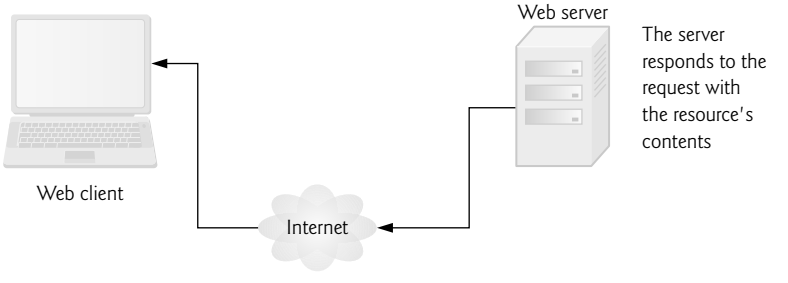


Fig. 1.9 | Client receiving a response from the web server.

The server first sends a line of text that indicates the HTTP version, followed by a numeric code and a phrase describing the status of the transaction. For example,

```
HTTP/1.1 200 OK
```

indicates success, whereas

```
HTTP/1.1 404 Not found
```

informs the client that the web server could not locate the requested resource. A complete list of numeric codes indicating the status of an HTTP transaction can be found at `www.w3.org/Protocols/rfc2616/rfc2616-sec10.html`.

HTTP Headers

Next, the server sends one or more **HTTP headers**, which provide additional information about the data that will be sent. In this case, the server is sending an HTML5 text document, so one HTTP header for this example would read:

```
Content-type: text/html
```

The information provided in this header specifies the **Multipurpose Internet Mail Extensions (MIME) type** of the content that the server is transmitting to the browser. The MIME standard specifies data formats, which programs can use to interpret data correctly. For example, the MIME type `text/plain` indicates that the sent information is text that can be displayed directly. Similarly, the MIME type `image/jpeg` indicates that the content is a JPEG image. When the browser receives this MIME type, it attempts to display the image.

The header or set of headers is followed by a blank line, which indicates to the client browser that the server is finished sending HTTP headers. Finally, the server sends the contents of the requested document (`downloads.html`). The client-side browser then renders (or displays) the document, which may involve additional HTTP requests to obtain associated CSS and images.

HTTP `get` *and* `post` *Requests*

The two most common **HTTP request types** (also known as **request methods**) are `get` and `post`. A `get` request typically gets (or retrieves) information from a server, such as an HTML document, an image or search results based on a user-submitted search term. A `post` request typically posts (or sends) data to a server. Common uses of `post` requests are to send form data or documents to a server.

An HTTP request often posts data to a **server-side form handler** that processes the data. For example, when a user performs a search or participates in a web-based survey, the web server receives the information specified in the HTML form as part of the request. `Get` requests and `post` requests can both be used to send data to a web server, but each request type sends the information differently.

A `get` request appends data to the URL, e.g., `www.google.com/search?q=deitel`. In this case `search` is the name of Google's server-side form handler, `q` is the `name` of a variable in Google's search form and `deitel` is the search term. The `?` in the preceding URL separates the **query string** from the rest of the URL in a request. A *name/value* pair is passed to the server with the *name* and the *value* separated by an equals sign (`=`). If more than one *name/value* pair is submitted, each pair is separated by an ampersand (`&`). The server uses data passed in a query string to retrieve an appropriate resource from the server. The server then

sends a response to the client. A `get` request may be initiated by submitting an HTML form whose `method` attribute is set to `"get"`, or by typing the URL (possibly containing a query string) directly into the browser's address bar. We discuss HTML forms in Chapters 2–3.

A `post` request sends form data as part of the HTTP message, not as part of the URL. A `get` request typically limits the query string (i.e., everything to the right of the `?`) to a specific number of characters, so it's often necessary to send large amounts of information using the `post` method. The `post` method is also sometimes preferred because it hides the submitted data from the user by embedding it in an HTTP message. If a form submits several hidden input values along with user-submitted data, the `post` method might generate a URL like `www.searchengine.com/search`. The form data still reaches the server and is processed in a similar fashion to a `get` request, but the user does not see the exact information sent.

Software Engineering Observation 1.1

The data sent in a `post` request is not part of the URL, and the user can't see the data by default. However, tools are available that expose this data, so you should not assume that the data is secure just because a `post` request is used.

Client-Side Caching

Browsers often **cache** (save on disk) recently viewed web pages for quick reloading. If there are no changes between the version stored in the cache and the current version on the web, this speeds up your browsing experience. An HTTP response can indicate the length of time for which the content remains "fresh." If this amount of time has not been reached, the browser can avoid another request to the server. If not, the browser loads the document from the cache. Similarly, there's also the "not modified" HTTP response, indicating that the file content has not changed since it was last requested (which is information that's send in the request). Browsers typically do not cache the server's response to a `post` request, because the next `post` might not return the same result. For example, in a survey, many users could visit the same web page and answer a question. The survey results could then be displayed for the user. Each new answer would change the survey results.

1.7 Multitier Application Architecture

Web-based applications are often **multitier applications** (sometimes referred to as ***n*-tier applications**) that divide functionality into separate **tiers** (i.e., logical groupings of functionality). Although tiers can be located on the same computer, the tiers of web-based applications often reside on separate computers. Figure 1.10 presents the basic structure of a **three-tier web-based application**.

The **bottom tier** (also called the data tier or the information tier) maintains the application's data. This tier typically stores data in a relational database management system (RDBMS). We discuss RDBMSs in Chapter 18. For example, Amazon might have an inventory information database containing product descriptions, prices and quantities in stock. Another database might contain customer information, such as user names, billing addresses and credit card numbers. These may reside on one or more computers, which together comprise the application's data.

The **middle tier** implements business logic, controller logic and presentation logic to control interactions between the application's clients and its data. The middle tier acts as

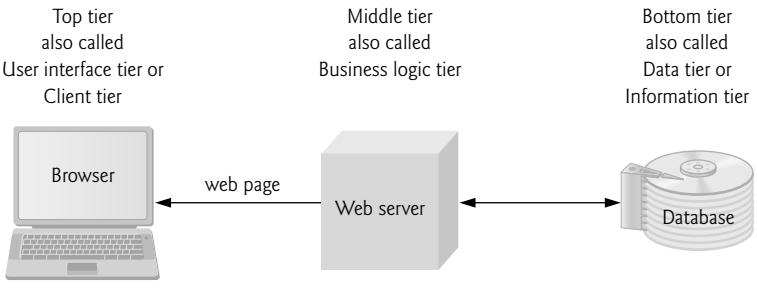


Fig. 1.10 | Three-tier architecture.

an intermediary between data in the information tier and the application's clients. The middle-tier **controller logic** processes client requests (such as requests to view a product catalog) and retrieves data from the database. The middle-tier **presentation logic** then processes data from the information tier and presents the content to the client. Web applications typically present data to clients as HTML documents.

Business logic in the middle tier enforces **business rules** and ensures that data is reliable before the application updates a database or presents data to users. Business rules dictate how clients access data and how applications process data. For example, a business rule in the middle tier of a retail store's web-based application might ensure that all product quantities remain positive. A client request to set a negative quantity in the bottom tier's product information database would be rejected by the middle tier's business logic.

The **top tier**, or client tier, is the application's user interface, which gathers input and displays output. Users interact directly with the application through the user interface, which is typically a web browser or a mobile device. In response to user actions (e.g., clicking a hyperlink), the client tier interacts with the middle tier to make requests and to retrieve data from the information tier. The client tier then displays the data retrieved for the user.

1.8 Client-Side Scripting versus Server-Side Scripting

Client-side scripting with JavaScript can be used to validate user input, to interact with the browser, to enhance web pages, and to add client/server communication between a browser and a web server.

Client-side scripting does have limitations, such as browser dependency; the browser or **scripting host** must support the scripting language and capabilities. Scripts are restricted from arbitrarily accessing the local hardware and file system for security reasons. Another issue is that client-side scripts can be viewed by the client by using the browser's source-viewing capability. Sensitive information, such as passwords or other personally identifiable data, should not be on the client. All client-side data validation should be mirrored on the server. Also, placing certain operations in JavaScript on the client can open web applications to security issues.

Programmers have more flexibility with **server-side scripts**, which often generate custom responses for clients. For example, a client might connect to an airline's web server and request a list of flights from Boston to San Francisco between April 19 and May 5. The server queries the database, dynamically generates an HTML document containing