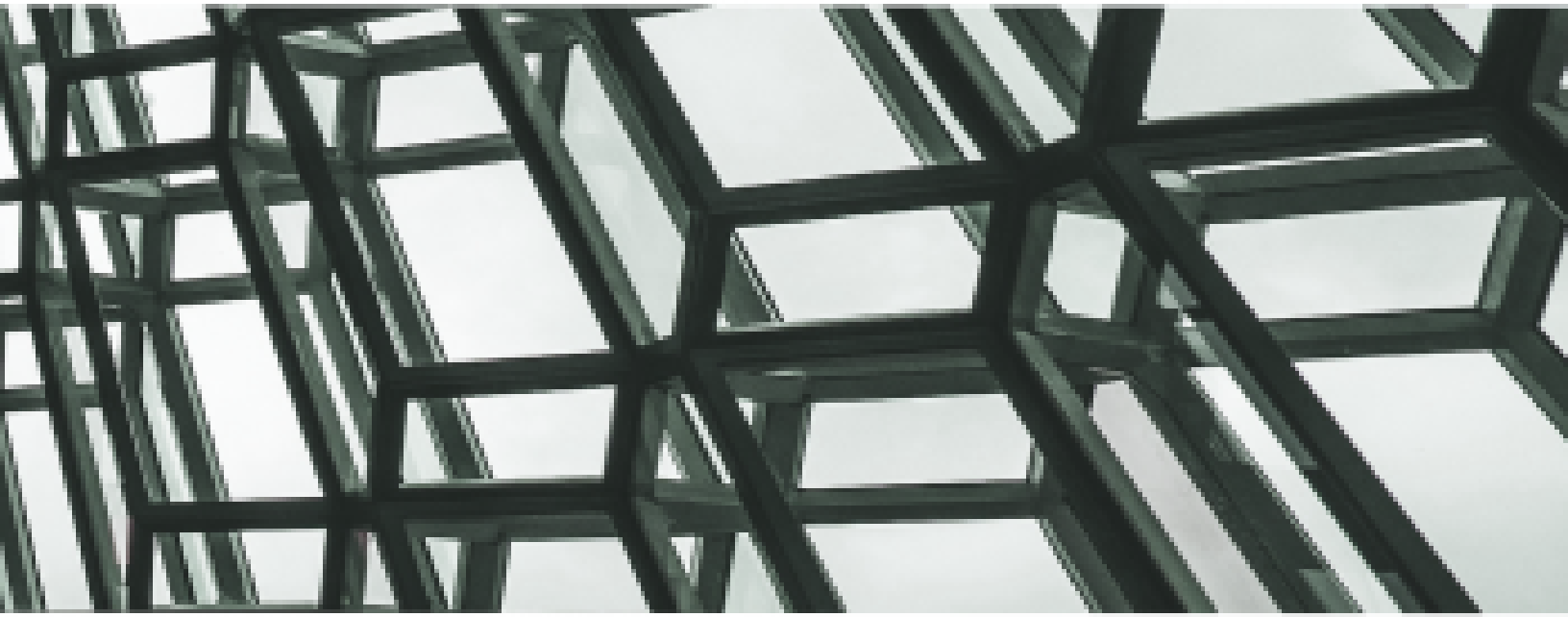
Web Design Primer

2018 EDITION



Ryerson University

Web Design Primer

Web Design Primer

RICHARD ADAMS & AHMED SAGARWALA

HUMAIRA IMTIYAZ, REUBEN KIBLITSKY, SHELLY SICAT

RYERSON UNIVERSITY
TORONTO

Contents

Introduction 1

Chapter 1 - Introduction 5
Chapter 2 - Files and Links 10
Chapter 3 - HTML 15
Chapter 4 - The Semantic Web 19
Chapter 5 - Styles and CSS 26
Chapter 6 - Image Optimization 36
Chapter 7 - Video 40
Chapter 8 - Animation 43
Chapter 9 – JavaScript 49
Chapter 10 – JavaScript Libraries 53
Chapter 11 - Uploading Content to a Web Server 59
Chapter 12 - eBook Formats 61
Chapter 13 - eBook Production 68
Chapter 14 - Digital Photography for Web and Cross-Media 73
Chapter 15 - Photoshop for Web and Cross-Media 89
Chapter 16 - Apps for iOS 99
Chapter 17 - References 106

Appendix - Image Credits 107

Introduction

Welcome to *Web Design Primer*!

This book was written for a one-semester course in web design for students in Ryerson's Faculty of Communication and Design (FCAD) and may also be useful for high school, community college, or training center courses on beginning web design — or for anyone who wants to read the book and complete the tutorials on their own.

The goal of the book is to provide students with a reference on some of the latest web design practices that is short and to-the-point, low-cost, and readily accessible.

What You Will Need

To use this book and its companion site for a course similar to Ryerson's, you will need:

- A computer, preferably with a large-screen display, for each student
- An HTML-editing program, such as Adobe Dreamweaver
- An image-editing program, such as Adobe Photoshop.
- An animation program, like Adobe Flash or Edge, Tumult Hype, or similar. If using Flash, Google's free "Swiffy" plugin enables exporting HTML5 that will work on iOS devices.
- Optionally, a server where students' work can be uploaded to and viewed on the web. In this case, a file-transfer utility like "FileZilla" may also be useful.

Acknowledgments

The authors would like to thank eCampusOntario, a not-for-profit centre of excellence and global leader in the evolution of teaching and learning through technology, for supporting and funding this project.

We would not have been able to complete it without the generous assistance of Dr. Wendy Freeman, Director of eLearning; and Tanya Pubuda, Ryerson eLearning Associate; along with librarians from the Ryerson Library,

including Ann Ludbrook, Copyright and Scholarly Engagement Librarian; Sally Wilson, Web Services Librarian; and Fangmin Wang, Head of Library Information Technology Services.

Our colleague Prof. Dr. Jörg Westbomke of Ryerson's exchange partner university, the Hochschule der Medien in Stuttgart, Germany, contributed material for the responsive design and search engine optimizations section of Chapter 4. Similarly colleague Prof. Jason Lisi of the School of Graphic Communications Management contributed material on non-destructive editing in Photoshop for Chapter 15. Thanks to Dr. Abhay Sharma for sharing his idea for the iPhone app, "Fruit Salad."

Thanks to the software publishers who granted permission to show screen captures of their programs in the book, including Adobe® Systems Inc. (Photoshop®, Dreamweaver®, and DNG Converter), Apple Computer (Xcode, iMovie, Safari), Tumult (Hype), The Mozilla Project (Firefox browser), and Google (Chrome browser). These software programs and their screen captures are subject to copyright by their respective publishers and are not part of the Creative Commons license that applies to this book [Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0)].

We are grateful to the students in the Masters in Digital Media program, class of 2019, who contributed material and support for the project, including Reuben Kiblitsky, Shelly Sicat, and Eleanor Beale; and Graphic Communications Management fourth-year student Humaira Imtiyaz, who also wrote material and helped design parts of the book.

About the Authors

Richard Adams is Associate Professor in Ryerson's School of Graphic Communications Management, where he teaches document design, web design, and color management. After completing his Ph.D. at Cornell University and working with scientific publications, he later returned for an M.S. in Printing Technology at Rochester Institute of Technology. He has taught at several universities and was a research scientist at the Graphic Arts Technical Foundation (now the Printing Industries of America). At Ryerson he completed the Certificate in Web Design from the Chang School of Continuing Education.

Ahmed (Am) Sagarwala is Manager of Industry Relations at Ryerson's Master of Digital Media, where he works with students in the graduate program. As an instructor, he teaches courses on digital literacy, design, and programming. He is also the president of Artform, a web technologies firm located in the Greater Toronto Area. He has his bachelor's degree from Ryerson University's School of Graphic Communications Management and a Master's in Digital Media. He is currently researching and developing a web energy disaggregation platform and loves ASCII art.

Chapter 1 - Introduction

So you want to learn how to design documents for the World Wide Web (www)? Well, before we get started it is important to learn why it was invented in the first place. Back in the year 1989, the Web was attributed to British scientist Tim Berners-Lee, who proposed it to facilitate communication between researchers. (Sir Timothy, who was knighted by Queen Elizabeth for his contributions to science, is now head of the W3C organization that sets standards for the web.)

Fast forward to present day, it serves as a primary tool to not only communicate, but also to entertain, collaborate and educate individuals. In this textbook, we aim to teach you the basics and fundamentals of how you can add value to the online space by building your own website.

Print vs. Web

We will assume you have some experience with page-layout programs for creating printed documents with text and graphics. With printed documents, the designer can specify the paper size and has control over the page dimensions. On the web, the designer has no idea which browser, display, or window size the reader will use. Analogous structures between printed and web pages are shown in **Table 1-1**. A basic web page is shown in **Figure 1-1**.

Table 1-1. HTML and analogous page layout elements

HTML Element	Analogy with Page Layout
DTD <DOCTYPE! html>	new page
inline element (letters, words, spaces, images, floated elements,)	spaced between
block element (<p>, <h1>, ,	return after

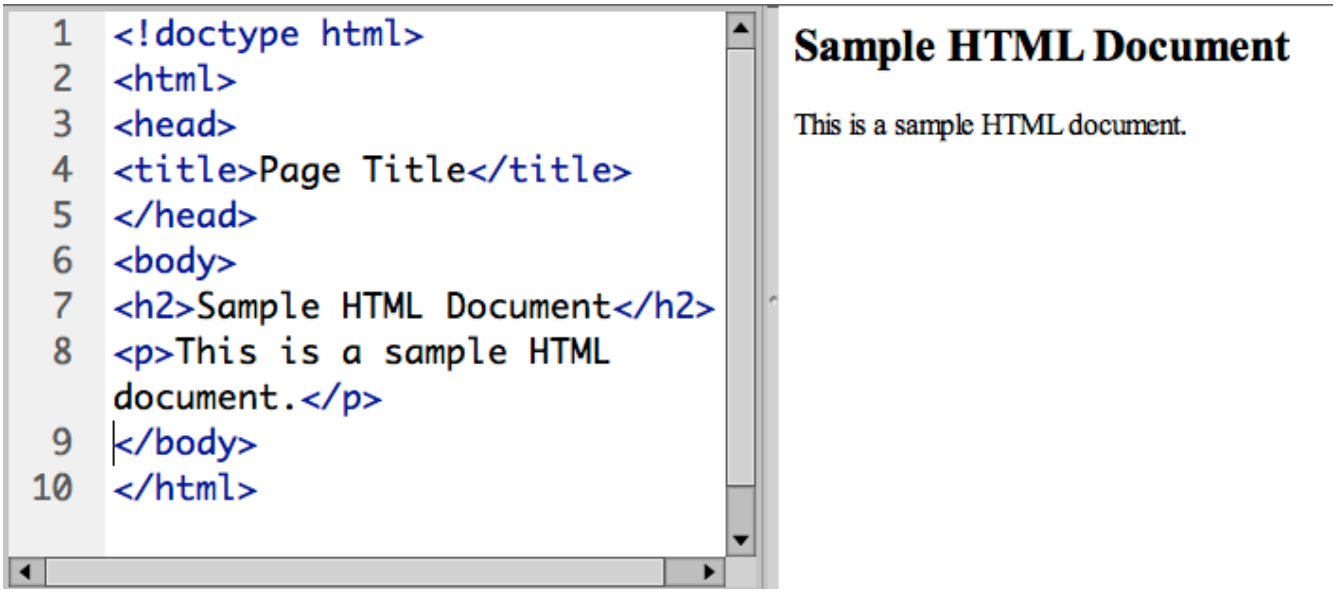


Figure 1-1. Structure of a basic HTML page for viewing with a web browser.

Inline vs. Block Elements

HTML elements are described as either inline or block, depending upon how they are arranged on the web page (**Table 1-2**). Inline elements fall into a series from left to right, like letters or words on a page (in a Western literary tradition)—like print elements with spaces in between. Block elements arrange themselves vertically on the page, like page elements with paragraph returns after them.

Incidentally, the “float” style can be used to convert block elements to inline, while the division tag can be used to make inline into block elements.

Table 1-2. Basic Page Elements

Element	What It Does
<!DOCTYPE html>	DocType Declaration (DTD) HTML5 document
<html>	Definition of the HTML page
<head>	Statements not visible to user
<title>	Title that appears in browser tab
<body>	Content visible to user

HTML, CSS, and JavaScript

These three components work together to create structured web pages with attractive styles and user interaction. HTML and its tags provide the structure of the page. CSS styles the tags, including size, colour, position, and many other characteristics. JavaScript adds interactivity, such as when users click buttons or submit forms.

The content and structure of a web page is defined by tags, abbreviations enclosed in less-than and greater-than symbols, like for paragraph. Most tags need to be closed using the same symbol, preceded by a forward slash, e.g., . Some tags are self-closing, such as image <img/>, line break
, and horizontal rule <hr/>.

Cascading style sheets (CSS) define the style of tagged elements, such as size, colour, position, margins, borders, and many other attributes. CSS refers to a separate file of type .css that contains the style specifications. The "cascade" also refers to the hierarchy of style specifications, which can be placed in any of three places:

1. Within the tag, or inline, such as
2. In the head of the document, or embedded, and enclosed with <style></style> tags and enclosed by braces {} after a selector that designates what tag the specifications are applied to
3. In a separate document of filetype .css

The hierarchy is that the first, or inline style, takes precedence over the embedded style, which takes precedence over the external .css file.

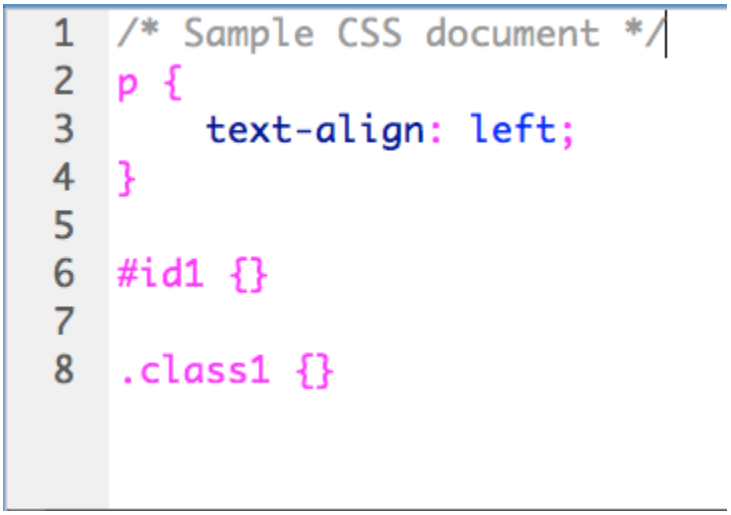


Figure 1-2. Structure of a style sheet with a selector ("p" for all paragraphs), ID (# symbol), and class (.).

Table 1-3. Text Elements

Element	Purpose	Default *Display* Value
	paragraph	block
<h1>…<h6>	headings (smaller number = higher emphasis)	block
	image	inline
	shape or box	block
	selection of text	inline
<article>	article content	block
<address>	mailing addresses and location details	block
	ordered list	numbered lines, block
	unordered list	lines with bullet points, block
	items in and	list items, block

Text

Text in web pages is indicated by the paragraph tag, . Paragraphs are block elements that line up after each other, as if the writer had put a paragraph return after each one.

Text can also include headings <h1></h1>, numbered from 1 to 6 (1 has the largest type, 6 the smallest).

Other text elements include <article> and <address> (Table 3).

Self-closing Tags

The following list contains special cases where tags don’t require a closing tag include. The trailing slash (/) is not necessary in HTML5, but often helps with tracking self-closing tags:

- The break
 tag, which creates extra vertical space between elements
- The horizontal rule <hr /> tag, which draws a horizontal line across the page
- The image <img /> tag, which inserts an image

CSS Selectors

Whether embedded in the head or written in an external .css file, cascading style sheets (CSS) work by applying styles to selected document elements, or selectors. An analogy from page layout would be when you select some text or a page element with the cursor and then apply a style to it.

Selectors consist of:

- Tag selectors, such as *body, div, p, h1*, or any other tag, written by itself

- IDs, named elements that are used only **once** on the page, indicated by a hash or number symbol (#)
- Classes, for multiple use, indicated by a period (.) before the name of the class

CSS Style Declarations

Style specifications, or declarations, in CSS are enclosed in braces or “curly brackets” (Figure 2). Each specification is generally listed on a separate line. The specification (e.g., “text-align”) is separated from the style (e.g., “left”) by a colon. Each declaration ends with a semicolon (;).

Comments

Comments are useful for reminding yourself how you have designed a page or sharing this information among a team of designers.

- HTML comments are enclosed in tags starting with an exclamation point and two dashes, and ended with two dashes:

  ```
  <!-- HTML Comment -->
  ```

- CSS and JavaScript comments start with a forward slash and asterisk, and end with an asterisk and forward slash:

  ```
  /* CSS Comment */
  ```

Designers often include dashes, equals signs, and asterisks in comments to separate sections.

JavaScript

The JavaScript programming language allows for interactivity on the web, such as actions triggered by buttons and animated effects. JavaScript statements are enclosed in opening and closing <script> tags. In addition to “plain” JavaScript written by the designer or programmer, libraries of preprogrammed JavaScript functions are available, such as jQuery.

Chapter 2 - Files and Links

Since a web page may have numerous linked files, like image and style sheets, and hyperlinks to other pages, file organization is very important.

A basic website will reside in a folder on a server that readers access through a universal resource locator (URL)—the web address that they type on their browsers. To load without being specifically named, the first page, or home page, on your site must reside in the main, or root, folder, and be called “index.htm” or “index.html.”

File Organization

A convenient way to organize a basic site is to make folders to hold the other web pages, images, style sheets, and JavaScript if present (**Figure 2-1**). Links (**Figure 2-2**) to these other elements need to be specified using the directory path on the server. For example:

- Going down: If you place an image, “image.jpg,” that resides in a folder, “images,” then you must specify <img src=“images/image.jpg”>, including the folder followed by a forward slash in the file specification.
- Going up: If in another page of your site, that’s contained in a folder, “HTML,” you want to use the same image, you must first specify an upward path from the “HTML” folder into the root folder, <img src=“../images/image.jpg”>. The two periods and slash tell the browser to go up one folder and then down into the “images” folder.

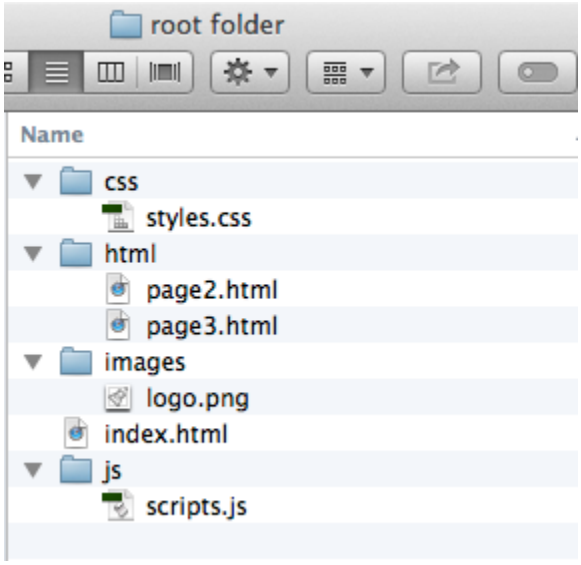


Figure 2-1. File organization of a simple web site into a root folder with an index.html file and folders for style sheets (css), other pages (html), and JavaScript (js).

Hyperlinks

Other pages can be linked using hyperlinks of two types:

- Relative hyperlinks are to pages on the same server, such as “page2.html” in **Figure 2-2**. Example of a relative link: Page 2.
- Absolute hyperlinks go to a link on a different server, usually someone else’s page on the web. An absolute link: Ryerson University.

```
1  <!-- Styles Link in Head  -->
2  <link href="styles.css" type="text/css" rel=
   "stylesheet">
3
4  <!-- JavaScript Link in Head -->
5  <link href="javascript.js" type="text/javascript">
6
7  <!-- Image Link in Body -->
8  <img src="image.jpg" />
9
10 <!-- Hyperlink to Another Page in Body -->
11 <a href="page2.html">Click Here for Page 2</a>
```

Figure 2-2. Examples of code for linking to embedded files and images.

Dreamweaver's Site Manager

Dreamweaver's Site Manager (Site > Manage Sites) provides a convenient way to manage sites (Exercises). The root folder is displayed in the Files palette (Window > Files, **Figure 2-3**). After creation and editing, files and folders can be uploaded to the server with a built-in file transfer protocol (FTP) utility.

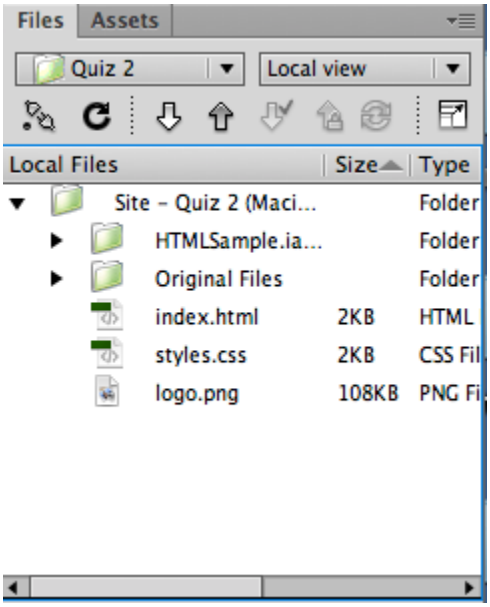


Figure 2-3. Dreamweaver's File palette.

Using Dreamweaver's Site Manager

How to use Dreamweaver's Site Manager

1. Make a site folder for your site and organize any desired subfolders.
2. Dreamweaver > Site > Site Manager.
3. New Site > enter Site Name > click the folder icon to the right of Local Site Folder.
4. Browse to the root folder you created, select, click Choose.

Linking to a Server

1. In Site Manager, click Servers.
2. Contact your site administrator for FTP settings to use.

Using the File Palette (Figure 3)

1. Open the File palette: Window > Files.
2. You can view various root folders on the local or remote computer (server).
3. To connect to the server, click the plug icon in the upper left corner.
4. If you change the file structure in the root folder, click the Refresh button.

Site Maps

A site map (**Figure 2-4**) is a diagram of the pages and hyperlinks in your site. It shows how readers will be able to navigate from one page to another by clicking on hyperlinks. A site map is useful in planning your site and determining the arrangement of files.

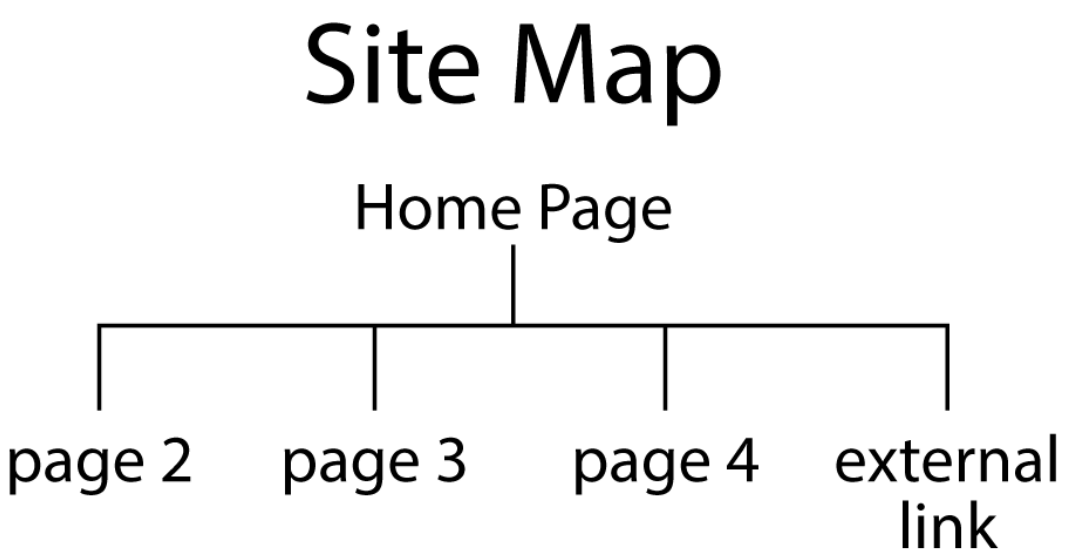


Figure 2-4. A simple example of a site map. Hyperlinks will enable the reader to navigate from the Home Page to pages 1, 2, and 3, and to an external website, by clicking links.

Chapter 3 - HTML

The structure of web pages is provided by the hypertext markup language (HTML), which includes various tags for text and objects. Tags are enclosed in less-than and greater-than symbols, such as the paragraph tag, . Most tags have opening and closing tags. The closing tag is written with a forwards slash (/) before the tag. Thus, we have a paragraph: Sample paragraph .

HTML is different from page layout for print because, while the print designer most likely knows the finished size of the document, and which is the same for all copies, the web designer does not know the type of equipment that the reader has — Is it a Windows or Macintosh computer? Android, iOS, or Windows tablet or smartphone? What size monitor? What size browser window? HTML code allows for the page to be created for each reader's equipment and settings.

Page Structure

Web pages are created with the <html> tag, which is analogous to a piece of paper in print. The area of the page visible to users is the <body>, analogous to the finished size of a print document. The <head> contains tags invisible to the user, such as <title> (name in the browser tab) and <style> (global or embedded style). Each web page should only have one <head>, and <body> tag.

Constructing a Web Page

To construct a web page, the designer would enclose each paragraph in opening and closing tags. Headers could be used to mark important sections, <h1> being the largest and <h2> the smallest.

The designer can style the type elements to include type style (font-family), type size (font-size), line leading or spacing (line-height), position (text-align), and colour (color).

Images in compatible file formats (JPEG, PNG, GIF) can be placed on the page using the <img> tag, e.g., . Unlike in page layout programs for print documents, images cannot be placed visually on the page. They can occupy their own lines or a text wrap can be created with the "float" style (e.g., <img src="filename.jpg" style="float: left;">). Space can be placed around the image using the "margin" style.

Relevant items can be grouped using the <header> and <footer> tags, usually at the top and bottom of the page, respectively, or with a tag in the main body.

Table 3-1. Common HTML Tags

tag	description
Text tags	
<p>	paragraph
<h1>…<h6>	heading 1 to 6 (1 is largest)
	unordered list, or bullet points
	ordered list, or numbered steps
	list items in a or series; can also be used to construct menus
Object tags	
<img/>	image, must be accompanied by “src” and the name of the image file; self-closing tag
	division, can act as a picture box, text box, grouping tag, or object unto itself
<a>	anchor, or link to another page, another location on a page, or another site. Accompanied by the name of the link as “href,” e.g., <a href=”filename.html”>
 	break, or carriage return; can be substituted with margin-bottom or padding-bottom; self-closing
<table>	table
<tr>	table row; each table needs at least one <tr> tag to appear
<td>	table data, or cell; each table needs at least one <td> tag to appear; columns are determined by the number of <td> tags in a row. Cells can be merged with the “colspan” tag, e.g., <td colspan=”2″>
<video>	inserts a video of compatible file type into the page
<audio>	inserts an audio clip of compatible file type
Grouping Tags	
<header>	area at the top of a page, e.g., could hold a banner and menu
<footer>	area at the bottom of a page
<nav>	navigation, useful for holding a menu of links to other pages or sites
	mentioned above, could also be used to group objects

The Four “Ws” of Web Design

When designing web pages, the authors recommend recognizing four design principles:

1. **Design for Width.** Although the web designer does not know the width of the user’s browser window, he or she can design for numerous possibilities by specifying width as percent and by implementing “responsive design” through the use of media queries (see Chapter 5). Whatever the width of the

browser window, users can scroll to access the page's height dimension.

2. **Who.** Although target audiences can be defined and/or assumed, we never know who will be viewing our web pages. Regardless, we want to welcome all appropriate readers and enable them to conveniently read the information on our site.
3. **What.** We do not know what equipment the user has, e.g., Macintosh or Windows computer, tablet, or smart phone; screen size; browser; installed fonts; internet connection speed; and other factors. Still, we try to create designs that look good on the widest variety of devices.
4. **Where.** We do not necessarily know where our readers came from, but we still want them to find our site appropriately and not to lose the link to our site. For example, when linking to another site, using the target="_blank" specification will cause the linked site to open in a new tab or window, thus preserving the reader's access to our site.

Links

A key feature of web pages is the ability to navigate among pages, locations within a page, and other sites. Links are created with the anchor tag, <a>, which is generally accompanied by "href" and the name of the page, location (ID), or site. Three types of links include:

- *local or relative links*—These go to other pages in a site. Format: <a href="filename.html">. The complete path to the file must be written in the tag. E.g., the above tag assumes that "filename.html" is present in the same folder as the current file. If it was in a folder called "pages," the tag should read <a href="pages/filename.html">
- *anchors*—Anchors to go to another location in the same page. The destination is represented by a tag with an ID, and the ID is cited in the anchor. E.g., <a href="para1″>. Anchors are often used to allow readers to conveniently navigate back to the top of the page or to an area further down on the page.
- *global or absolute links*—These go to other sites, e.g., <a href="http://www.ryerson.ca">. Again, specifying target="_blank" will open the linked site in a new browser tab or window, preventing readers from losing our site.

Forms

"Here, fill out this form!"—not a pleasant sound to many people's ears. Adding to that is the concern about seeing the fine print, that someone can't read your handwriting, and those repetitive fields. Thanks to the web, forms can now be created for users to fill in online—perhaps in advance.

A form can be written on the web or given to end-users to download as Fillable PDF and either upload, email, or submit the forms data. An advantage of PDF is that the user can save the form, change the data, and submit it again if required.

Online forms can use multiple types of objects for collecting data, including text fields, radio buttons, check boxes, and selection lists (**Table 3-2**).

Table 3-2. Form Field Types

Purpose	Description	Code
text and numerical entries	fields for readers to type information	<input type="text">
radio buttons	selecting a single option from a predefined list	<input type="radio">
check box	specifying one or multiple items as present or absent	<input type="checkbox">
text areas	allows for the entry of multiple lines of text	<textarea>
selection lists	lists of options in drop-down lists	<select> <option>
submit button	transmits form data	<input type="submit">

The form part of a web page is designed by the <form> tag. The form tag should include a name, ID, action, and method.

Chapter 4 - The Semantic Web

Semantics is a branch of linguistics which focuses on meaning. When web designers talk about a semantic web, it's a conversation about the use of HTML tags and structure to convey meaning. HTML5 introduces a plethora of tags that can be used to structure content in order to make it meaningful to a machine. Remember that web languages are used not just by users, but also by search engines who rely on structured content to generate search results. Assistive technologies such as screen readers need tags to express content. SEO and accessibility are therefore enabled through semantic HTML tags.

Web designers are challenged to not only architect visually intriguing sites, they must also consider the user's technology and abilities to navigate a digital medium. Therefore, the questions we should ask before getting started are: who are my users? What do they want to access? How will they access this content? Where could they be when accessing the content?

We could define our user as any human being. The problem with this is that it's too vague. People come in all shapes and sizes. They have a variety of experiences and decision making processes. Just as their characteristics vary, their devices do to. Let's illustrate a few examples of what may vary:

User #1

- Visually impaired
- Relies on a screen reader to consume content
- Uses a keyboard to navigate page links

User #2

- Colour blind
- Uses a retina display with a resolution of 2,880 pixels by 1,800 pixels
- Connects using a high-speed (over 10-megabit) connection

User #3

- No vision problems
- Uses a screen with a resolution of 1,280 pixels by 720 pixels
- Connects to the internet using a dial-up connection

User #4

- No vision problems
- Uses their mobile device, a resolution of 800 pixels by 600 pixels
- Has 3G speeds, but finds data costly

Each of the users has a specific challenge to accessing content. So the question arises: how can we cater to everyone equitably? Luckily, the W3C (Word Wide Web Consortium) has been addressing such challenges for decades. Providing we are aware of their guidelines, we can create websites that are both accessible and engaging. Begin every web project with accessibility in mind. Think of your content first, rather than focusing on visual appeal. The approach this book will use is based on progressive enhancement. It's like dressing your content up—remember that you need to start with the body. Here's a basic workflow:

1. Organize your text in a document
2. Apply semantic HTML tags
3. Add images and create links as needed
4. Ensure images and links are described using *alt* and *title* attributes
5. Apply styles to text and format content
6. Use styles to position content
7. Integrate JavaScript for interactivity and changing browser behaviour

Semantic Code Example

Using the following content, here is a walk-through of how it can be optimized.

```
Navigation: Home, About, Contact
Heading: Our First Page
Body: Building an accessible website is as easy as structuring content with the appr
```

We don't need to include the words *navigation*, *heading*, and *body*. They've been included to classify the content to you. Many websites use *div* tags to structure content. A *div* tag has no meaning on its own. HTML5 provides semantic tags such as *header*, *footer, nav, section*, and *article.* As long as they're used appropriately, we can describe our content to a machine for interpretation.

```
<nav>Home, About, Contact</nav>
<header>Our First Page</header>
<p>Building an accessible website is as easy as structuring content with the appropr
<img src="images/sample.jpg">
```

It's a start, but we still need to add links to our navigation using an unordered list. An image has also been added, but you have no idea what is contains based on the filename. We can also nest tags to describe a block of content. Nesting is truly one of the best features of HTML. Here's how we can apply this to our code:

```
<nav>
  <ul>
    <li><a href="index.html" title="Head to homepage">Home</a></li>
    <li><a href="about.html" title="About this website">About</a></li>
    <li><a href="index.html" title="How to contact us">Contact</a></li>
  </ul>
</nav>
<article>
  <header>
    <h1>Our First Page</h1>
  </header>
  <p>Building an accessible website is as easy as structuring content with the appro
  <img src="images/sample.jpg" alt="Image of an accessible web page">
</article>
```

In the above example, a title attribute has been added to the links to describe them. The *ul* tag specifies the navigation as an unordered list. A screen-reader can interpret each link as a unique item and would allow a user to tab through each link. The *article* tag is used to keep the body of the page distinct from the navigation and any other articles that may be displayed. The *alt* attribute of the image (*img*) now lets us know it depicts an accessible website. This is quite helpful to anyone using a screen-reader. It also allows a search engine to classify it based on the keywords used in an *alt* attribute.

Comments on Flash and Similar Technologies

Macromedia created Flash in 1996 due to the lagging support of interaction on web pages. This technology was not created with accessibility as a priority. The goal was to dazzle web users with imagery and motion, and content was generally secondary to the visual approaches used by developers. In order to support Flash, users needed to install a proprietary plugin that was heavy (large in terms of size). Users with slower internet connections would often be left out because download speeds wouldn't allow for the plugin to be installed or updated easily. Sites running Flash would often not be optimized and therefore take a lot of time to load.

Data requirements aside, Flash as well as Microsoft's SilverLight are also closed source technology. Both are often used to transfer rights managed content to systems, making it difficult to save content for redistribution.

Closed technologies have their use-cases, but improvements to open standards-compliant web languages (HTML, CSS, JavaScript) means we no longer have to rely on Flash. HTML5 allows for streaming content, and CSS and JavaScript, when used together, enable the animation and interactivity once only afforded by Flash.

Responsive Design

The goal of responsive design is to create web pages that adapt to varying screen sizes. A page can look one way on a 21-in. iMac and still be easy to use on a 4.7-in. iPhone. Responsive design can be accomplished using two tools:

- Specifying width in % rather than px, which enables the objects to be resized according to the width of the reader's browser window.
- Using a media query like "@ media only screen and (max-width: 500px) {}" to specify styles for the smaller screen(s).

The media query can be placed in a global/embedded style statement in the head of a document or in a separate, linked CSS style sheet. The media query is positioned like a selector and has its own set of sub-selectors and style statements that apply to the smaller format.

Checklist for Responsive Design

The authors suggest the following considerations for responsive design:

- Menu and buttons
    - menu position, e.g., horizontal on large screen, vertical on small screen
    - :hover — change style when mouse is over tag
    - :active — change style when tag is clicked
- Page layout
    - change image display from text wrap (float) to block
- Change colour (color, background)
- Change contrast
- Change font
    - size (font-size)
    - leading (line-height)
    - bold
    - contrast

    - black/white vs. white/black

- Alt tag — use to describe images

Viewing Your Responsive Design

The web authoring program Dreamweaver, and most popular browsers have a “responsive design mode” or view (**Table 4-1**, **Figure 4-1**) for viewing your page as it would appear on various smartphones and tablets.

Table 4-1. Responsive Design Modes in Browsers

program or browser	how to enter responsive design mode
Dreamweaver	In Live View click the browser window size popup at the bottom of the screen
Safari	Preferences > Advanced > check Show Developer Menu Developer > Enter Responsive Design Mode > select device
Chrome	View > Developer > Developer Tools > click on Responsive Design button
Firefox	Tools > Web Developer > Responsive Design Mode > select device

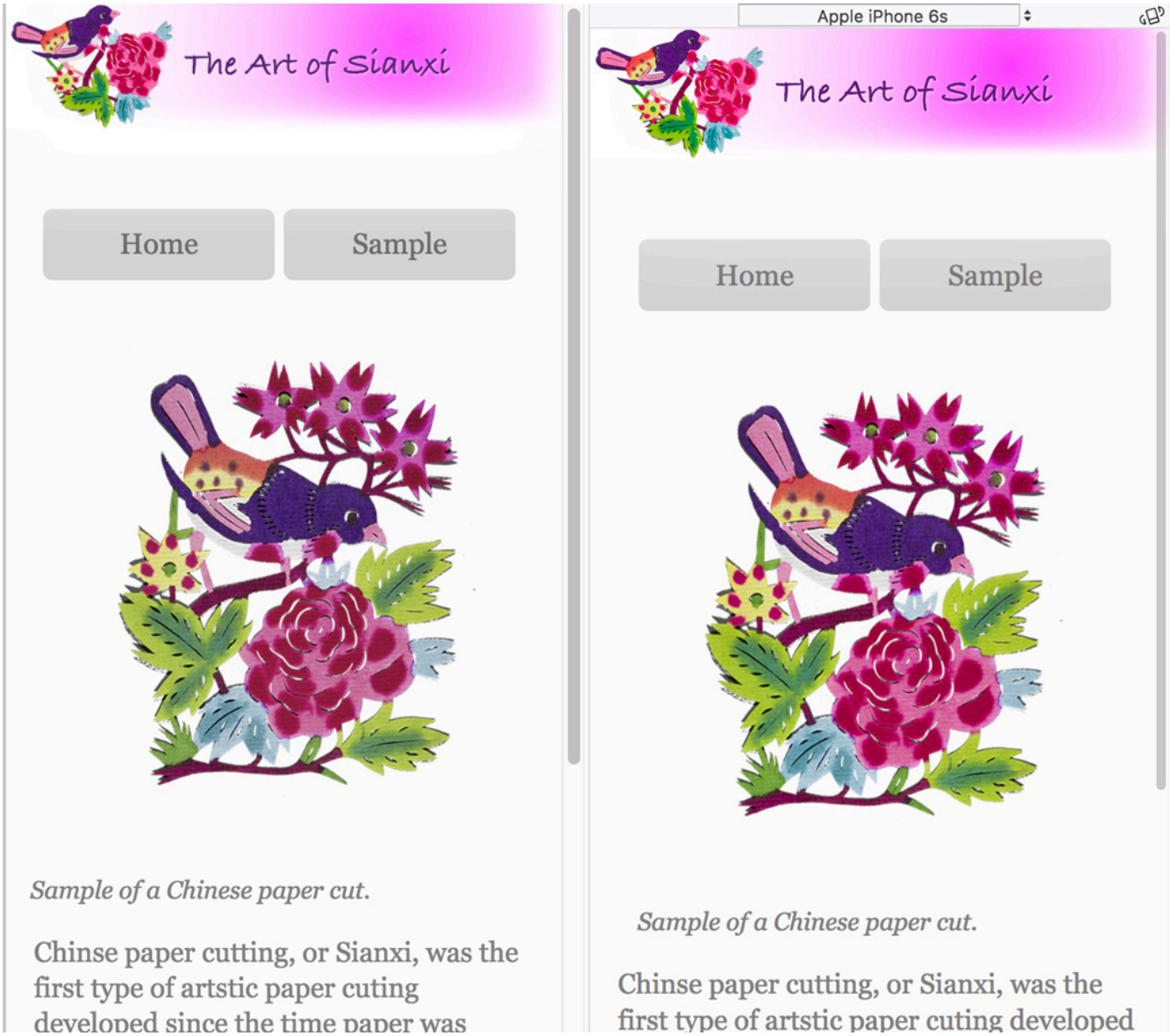


Figure 4-1. Page viewed in Dreamweaver's Live View (left) and Mozilla Firefox (right) in responsive design mode with iPhone 8. (Adobe® Dreamweaver® screenshot(s) reprinted with permission from Adobe Systems Incorporated.)

Search Engine Optimization

According to German market research firm Statista, in 2015 there were close to a billion websites in operation, and users made over 3 billion searches using the Google, Yahoo, Big, Baidu, and Yandex search engines.

Search engine optimization (SEO) refers to the inclusion of information in your site that makes it easy for a search engine's web crawler to find the site. According to Prof. Dr. Jörg Westbomke from the Hochschule der Medien in Stuttgart, Germany, SEO includes two types of optimization: on-page and off-page. On-page optimization includes steps that web publishers can take to increase the visibility of their sites to search engines. Off-page optimization refers to external factors, including links to a site from other sites, bounce factor, points in social media.

Checklist for Search Engine Optimization (SEO)

- Correct any spelling errors in the text.
- Add a title in the head of each page, <title>Title Goes Here</title>
- Add a meta tag with a description of the site, <meta name=”description” content=”description goes here”>.
- Add a meta tag with keywords, <meta name=”keywords” content=”keywords go here”>.
- Add “alt” tags describing all the images, .
- Rename the html pages (and rewrite links) with a descriptive URL.
- Add some headings <h1> that describe the text.
- Emphasize important words (that you want the web crawler to pick up) with emphasis and bold tags.

Chapter 5 - Styles and CSS

Cascading style sheets (CSS) are used to specify the appearance of web pages written in HTML. Appearance includes colour, dimensions, position, and behaviour of elements. The word “cascading” has a dual meaning: (1) the hierarchy of specifications, and (2) a file of type .css.

CSS file syntax is specified by the W3 Consortium. The file type was first proposed by Dr. Håkum Wium Lie, a web developer from Norway who was formerly the chief technology officer of Opera Software, publishers of the Opera browser.

The purpose of this chapter is not to be a comprehensive reference on CSS, which is available online at W3Schools.com and other sites, but to explain and give examples of how to use CSS.

Types of Style Sheets

CSS includes three different ways of specifying styles that are recognized in a hierarchy from greatest to least influence. **Figure 5-1** shows examples of all three:

- *inline* or *local* styles are written right in the HTML tag and take precedence over other styles. Example: <p style=”font-family: Arial; color: dimgray”>. The advantages of the inline/local style are that it is easy to write, easy to understand its effect on the corresponding tag, and overrides other styles. The disadvantage is that writing the same style for multiple tags could be laborious.

- *embedded* or *global* styles are written in the head of the document and apply to the entire document. Embedded styles are overridden by inline/local style sheets but not by external CSS files. Embedded/global styles are written inside opening and closing <style> tags, and the specifications are enclosed in braces {}. The advantage of the embedded/global style is that it is easy to write and see the effect of the specifications since the style is at the top of the same document it controls. The disadvantage would be that the global/inline style statement would need to be copied to all pages that use the same styles.

- *external CSS* files are files of type .css that are linked to a document using the link tag (e.g., <link href=”styles.css” rel=”stylesheet” type=”text/css”>. External CSS do not need a <style> tag and simply include the styles enclosed in braces. External CSS files are overridden by both embedded/

global styles in the document head and inline/local styles in the tags. The advantage of the external CSS file is that it can be linked to and set the styles for more than one page.

```
<!-- External/linked CSS file -->
<link href="styles.css" rel="stylesheet" type="text/css" />

<!-- Global/embedded style statement -->
<style>
    p {
        font-size: 24pt;
    }
</style>
</head>

<body>
<!-- Local/inline style -->
<p style="font-family: Helvetica;">Example of 3 “cascading style sheets.”</p>
</body>
</html>
```

Figure 5-1. This Dreamweaver code view shows examples of the three types of cascading style sheets, labeled with comments (grey text).

Styling Specifications

CSS specifications can include dimensions, position, colour, and background colour of objects. Specifications can only be made if they can be applied to the specified tag, or object. For example, consider a paragraph tag, <p>, and a division tag, , which could be used as an object or grouping tag. The style “color,” when applied to the paragraph, will set the colour of the text. However unless the division contains text, the colour style is not relevant to the division itself. The style “background” or “background-color,” when applied to the paragraph and a division, will colour the background of both.

Some common style specifications are listed in **Table 5-1**.

Table 5-1. Common Style Specifications

Use	CSS Property
color	color (text color)
	background-color (background)
dimensions	width
	height
position	static
	relative
	fixed
	absolute
	sticky
font	font-family
	font-size
	font-weight
	font-style
	text-align
	text-decoration
	line-height
float	left
	right
	none
display	inline
	block
spacing	margin
	padding

How to Write Styles

Inline/Local Styles

1. Inside a tag, such as a paragraph, <p>, write style="", placing the style specifications inside the quotation marks. (Use vertical quotation marks rather than left- and right-handed quotation marks.)
2. Follow each attribute with a colon, and end the specification with a semicolon.
3. Example: <p style="font-family: Arial, Helvetica, sans-serif; font-size: 12pt;">

Embedded/Global Styles

1. In the head of the document, place opening and closing <style> tags.
2. Inside the style tags, write selectors (the names of tags, without the less-than and greater-than symbols), open a brace {, write the attribute with a colon and the style specification ending in a semicolon, then close the brace }.
3. Designers customarily begin the next selector tag with the closing brace from the previous tag and then open the succeeding brace, so the tag looks like it is surrounded by backward braces. This makes the style specifications easier to read.

Exercise

Separate CSS File

1. Create a file of type .css.
2. Inside the file, begin writing styles as per the instructions for embedded/global styles, but without the <style> tag.
3. Save the file in a location relative to the HTML file(s) that it will control.
4. Link each HTML file to the .css file using the code <link href="styles.css" rel="stylesheet" type="text/css">
5. The above link assumes that "styles.css" is in the same folder as the HTML file that links to it. If the "styles.css" was in a folder called "styles," then the link should specify the folder path and read "styles/styles.css."

Selectors—Tags, IDs, and Classes

Global/embedded styles and external style sheets require that the tags to which specifications are applied be selected, meaning that the CSS has to know which tags to apply the styles.

Selectors can include tags, IDs, and classes, along with pseudo-classes and pseudo-selectors (**Table 5-2**).

- *Tags*—To use a tag as a selector in global/embedded or external CSS, simply type the tag inside the <style> statement in the head or in the external .css file, followed by the style specifications enclosed in braces. Tag selectors apply to all tags in the page, unless overridden by an ID, class, local/inline style, or "!important" specification.
- *ID*—To assign an ID, add ID="name" inside the tag. An ID can only be used once in a page. Select the class in CSS using "#name."
- *Class*—To assign a class, add class="name" inside a tag, then select in CSS using a period before the name, e.g., ".name."

Table 5-2. Selectors

selector	examples	description
tag	<p> p {styles}	applies to all occurrences of the tag
ID	<p ID=”para1″> #para1 {styles}	can only be used once within a page, e.g., for one paragraph, one heading, one image, etc.
class	p class=”para1″ .para1 {styles}	can be used multiple times within a page

Colour in CSS

Colours can be specified using one of four systems: name, RGB, RGB-A, hexadecimal, or abbreviated hexadecimal. Of these systems, named colours are easiest to understand. RGB can be used to specify a wide variety of colours in a way that is also understandable. RGBA offers the option of opacity.

- *name* refers to the 140 named colours on the web, such as aliceblue, dimgray, or chartreuse. Examples: color: dimgray; background: aliceblue;
- *RGB* refers to red, green, and blue values on a scale of 0–255, where 0 is black, or no colour, and 255 is maximum colour. Example: color: rgb(127,127,127) would be grey.
- *RGBA* includes an additional specification for opacity, on a scale of 0 to 1, with two-decimal accuracy. Example: color: rgba(127,127,127,0.50).
- *hexadecimal* includes colour values designated with six characters on a scale of 0 to 16. Digits above 9 are represented by letters A–F. Example: 0000FF is blue.
- *hexadecimal short* uses six characters to designate colours. Example 00F is blue.

Float and Clear

The float style can be used to create a text wrap, i.e., a photo with text wrapped around it. The styles “float: left” or “float: right” would be applied to the photo. (There is no “float: center.”) The styles “float: none;” and “clear: both;” remove the float from succeeding items. However, clear maintains its effect throughout the document.

Spacing

Spacing inside and around objects can be set with the padding and margin styles (**Figure 5-2**). Padding refers to space inside an object, e.g., a <p> paragraph tag containing text could have “padding” around the text and “margin” around the outside of the tag.

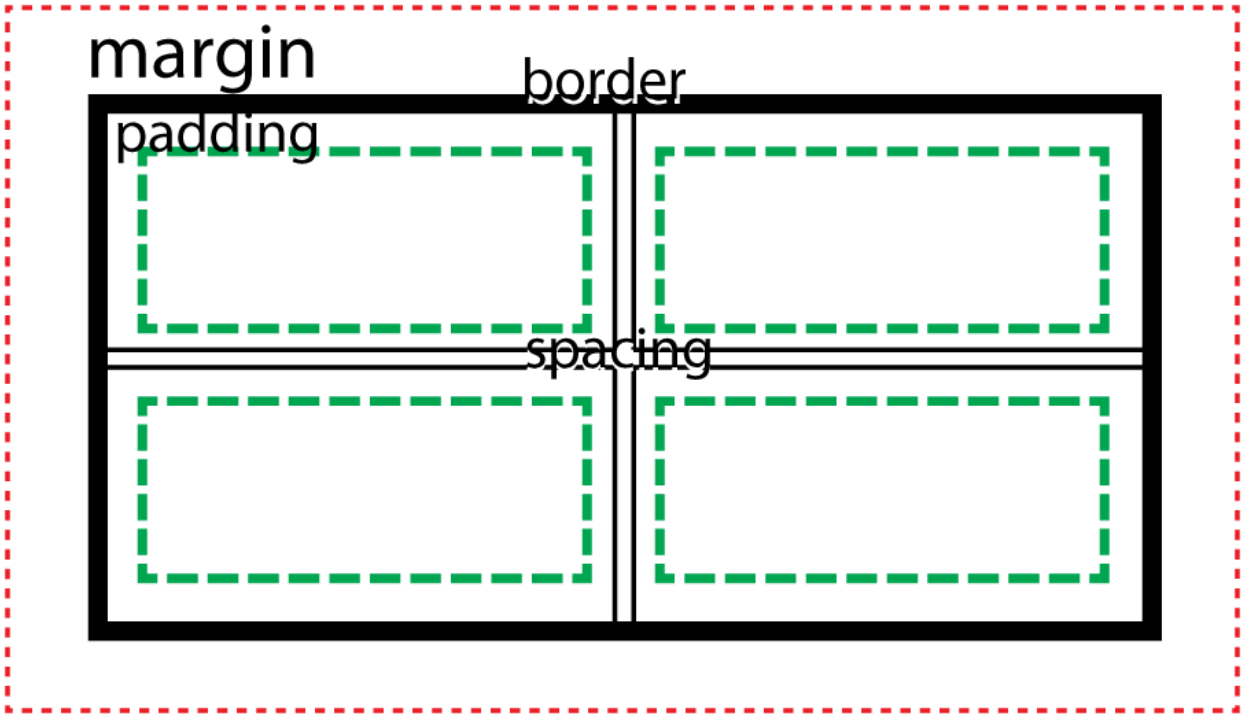


Figure 5-2. A table with two rows and four cells are used to illustrate the concepts of margin (red line outside the table) and padding (green line inside each cell).

Spacing is specified in pixels or other units and one, two, or four values:

- One value means spacing is the same on all sides.
- Two values represent top/bottom and left/right.
- Four values specify space from the top in a clockwise direction (top, right, bottom, left).

One way of centering an object is to use the style “margin: 0 auto,” where “auto” refers to automatic left and right margins (Figure 0). For “auto” to work, the width of the object must be specified.

Position

Styles for the position include “static”, “relative”, “absolute”, “fixed”, and “sticky”. Some position styles useful for specific purposes include:

- sticky—the element stays in place. This is used for menus and other references that the designer wants to remain visible to users after scrolling.
- absolute—the element remains in the same location relative to a container that it’s in.
- fixed—the element remains in a set location, defined by coordinates from the top or bottom and left or right of the window. This is useful for creating anchors that for example allow readers to return to the

top of the page.

Pseudoclasses for Interactivity

Pseudoclasses are built into HTML and do not need to be defined by the web designer. They define four states of mouse or pointer interaction, usually with links. The syntax is selector:pseudoclass {styles}, e.g., a:link {}.

- :link—defines the appearance of a tag when it has not yet been clicked. E.g., the default appearance of links <a> is blue type with an underline.
- :hover—appearance of the tag when readers hover the mouse over it. This pseudoclass is valuable in providing feedback that the user is in the correct location to click a link.
- :active—appearance of the tag when clicked. Again this could be a followup to the :hover pseudoclass by providing feedback that the link has actually been clicked and some action should follow.
- :visited—appearance of the tag after it has been clicked. Again, the default appearance of links that have been visited is purple text with underlining.

"Child" Pseudoselectors

So-called pseudoselectors are built into HTML and provide an efficient way of selecting multiple sub-elements that are part of a tag, such as rows <tr> in a table <table>.

Table 5-3. Pseudoselectors

selector	description
:nth-child(1)	selects first "child" in a series
:nth-child(odd)	selects odd-numbered "children" in a series
:nth-child(2n+3)	selects every second "child," starting with the third

Creative Effects in CSS

- Gradient—CSS can include gradient backgrounds (**Figure 5-3**). The gradient can be linear or radial, include two or more colours, and have a direction specified in terms of right/left, top/bottom or at an angle.

Figure 5-3. Example of a radial blend applied to a div enclosing an image. The syntax used was “background: radial-gradient(plum, white, white);”

- Drop shadows—These can be applied to text (text-shadow) or to rectangular items (box-shadow, **Figure 5-4**). Specifications include the horizontal offset, vertical offset, blend distance, and shadow colour. Example: text-shadow: 3px 3px 6px gray;

Figure 5-4. Two examples of drop shadows, one applied to text (text-shadow) and another to the paragraph (box-shadow).

- Flexible box layout—The “display: flex” and accompanying style specifications can be used to determine the distribution of multiple objects on a page. A common use is to center items horizontally and/or vertically (**Figure 5-5**).

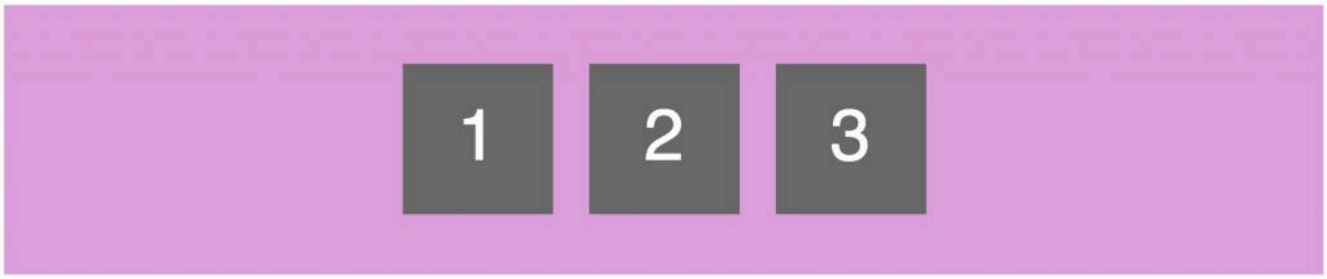


Figure 5-5. The “display: flex” style was applied to the plum-coloured div, and the enclosed paragraphs 1, 2, and 3 were aligned by setting the div to “justify-content: center” (horizontal) and “align-items: center” (vertical).

- Viewport width (vw) and viewport height (vh) measurements—These scalable measurements (**Figure 5-6**) take the place of fixed measurements like pixels (px) and are great for responsive design. One vw is equivalent to 1% of the viewport width. The vw and vh units can be used to size tags, type, leading, margin, and padding.

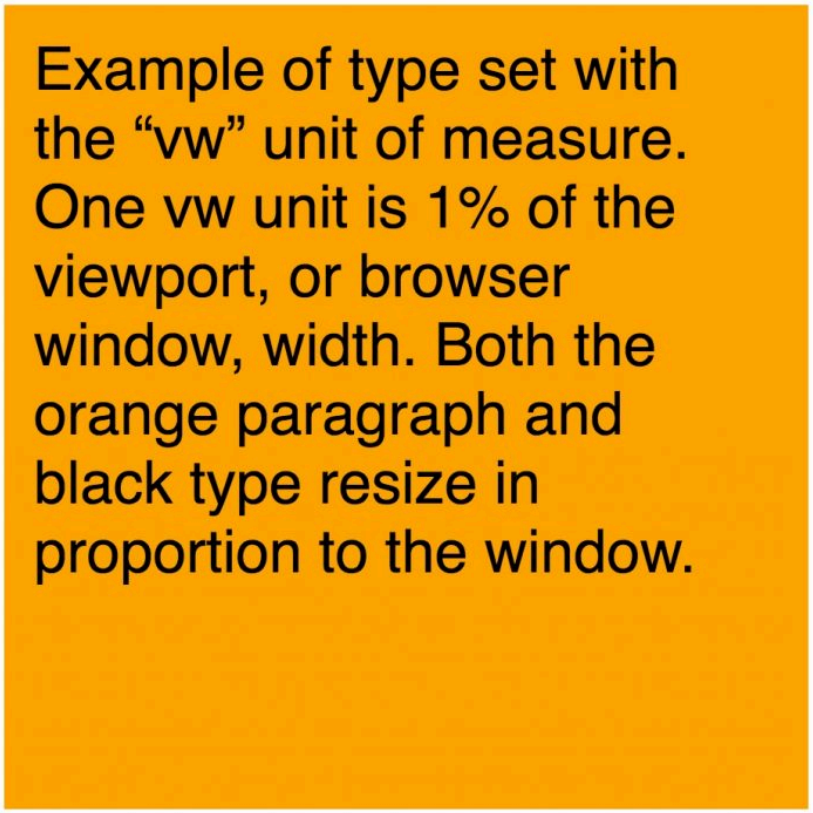


Figure 5-6. Both the paragraph (orange background) and type were measured using vw units. Both scale with the size of the browser window.

- Drop caps—Large first initials (**Figure 5-7**) can be created by selecting the first letter of a paragraph, either by using a tag around the letter or by using the pseudo-element ::first-letter (e.g., p::first-letter). Then specify font size in percent or em, line-height to get the desired vertical position, colour, and margin.

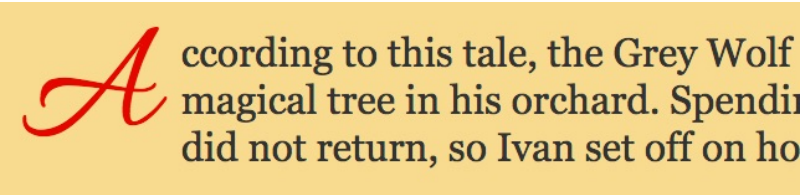


Figure 5-7. Drop cap created with a::first-letter pseudo-element, larger type, line height, margin, and colour.

Variations in Web Browsers

Although the latest versions of web browsers conform to W3C standards and interpret HTML and CSS the same, some differences do exist, particularly with newer and more complicated styles and in the way they are handled by older browser versions. Styles may need to be written differently for browsers, depending upon their underlying page layout engine (**Table 5-4**).

Table 5-4. Major Web Browser Engines

Engine	Browser	Abbreviation
Trident	Internet Explorer	
WebKit	Safari, Chrome, Opera	-webkit-
Gecko	Firefox	-moz-

An example of CSS code for different browsers, as applied to the CSS animate style, is:

```
-webkit-animation-name: example; /* Safari 4.0 - 8.0 */
-webkit-animation-duration: 4s; /* Safari 4.0 - 8.0 */
animation-name: example;
animation-duration: 4s;
```

Chapter 6 - Image Optimization

This chapter provides a brief introduction to images for the web. More details are given in Chapter 14, Digital Photography for Web and Cross-media, and Chapter 15, Photoshop for Web and Cross-media.

The most recent versions of web browsers can read RGB images in JPEG, GIF, PNG-8 and -24, and SVG (**Table 6-1**). Images for web sites should be saved at final size and 72 ppi resolution (except SVG which is resolution-independent).

Table 6-1. Images Supported by Browsers

Format	Compression	Type	Transparency	Description
JPEG	variable	contone	no	Joint Photographic Experts Group, continuous tone, multiple compression levels
GIF	variable	indexed	yes	Graphic Interchange Format
PNG-8	none	indexed	yes	Portable Network Graphic, 8-bit, indexed color
PNG-24	none	contone	yes	Portable Network Graphic, 24-bit, continuous-tone
SVG	none	indexed	yes	Scalable Vector Graphic, line drawings and flat artwork

Web vs. Print

If you're used to working with images for print, you probably know that their resolution should be twice the screen ruling, or 300 ppi for a 150 lpi halftone screen. For those without a printing background, presses can only print solid ink. To reproduce properly, images must be broken down into nearly invisible dots. The screen ruling describes how many dots there are per linear inch.

If you work with page-layout programs like Adobe InDesign, you also know that you should size images before placing them in pages; but you can also adjust the size in InDesign, as a percentage of the original size.

Images for the web should also be sized for correct reproduction. Although they could be resized in Dreamweaver or by using the height and width styles, resizing will not produce the optimum file size. That is, sizing them smaller will make the resolution too big. Resizing larger may reduce the quality by producing a pixelized or stair-stepped image. Optimum file size is important because it speeds up page loading.

Web Image Formats

Continuous-tone. Continuous-tone, or “contone”, refers to bitmapped images that are photographs or drawings with many shades and blends of colour. Contone images are best saved in JPEG or PNG-24 format (**Figure 6-1**).

***Figure 6-1**. The same image is saved in TIFF, JPEG, and PNG-24 formats (then combined in JPEG format for this figure).*

Flat colour. Flat, or indexed, colour refers to bitmapped images of drawings and line art that have a limited number of colours (**Figure 6-2**). Examples are comic strips and logos. Flat colour is best suited for GIF or PNG-8 format.

***Figure 6-2.** The same image is saved in GIF and PNG-8 format.*

Vector graphics. Vector graphics include resolution-independent drawings and diagrams drawn in Adobe Illustrator, CorelDraw, and similar programs where points, lines, curves, and shapes, along with stroke and fill, are defined by mathematical equations (**Figure 6-3**). Vector graphics can be saved in SVG format for display on the web. SVG defines vector graphics in XML format. Linked SVGs are displayed in Dreamweaver as a separate tab with the XML code.

Browsers cannot display print-based vector graphic formats, including Adobe Illustrator (.ai) and encapsulated PostScript (EPS). Adobe portable document format (PDF) files can be viewed with browser plugins but display as separate documents.

SVG files can be imported into HTML pages using the same <img> tag as for bitmapped images.

Adobe Illustrator has an Export > Save for Screens dialog box similar to Photoshop's and includes the option to save vector graphics in SVG, along with PNG, JPEG, and GIF bitmapped formats. Designers should consider whether scaling will be necessary, given scaling of the graphic will be limited by screen size.

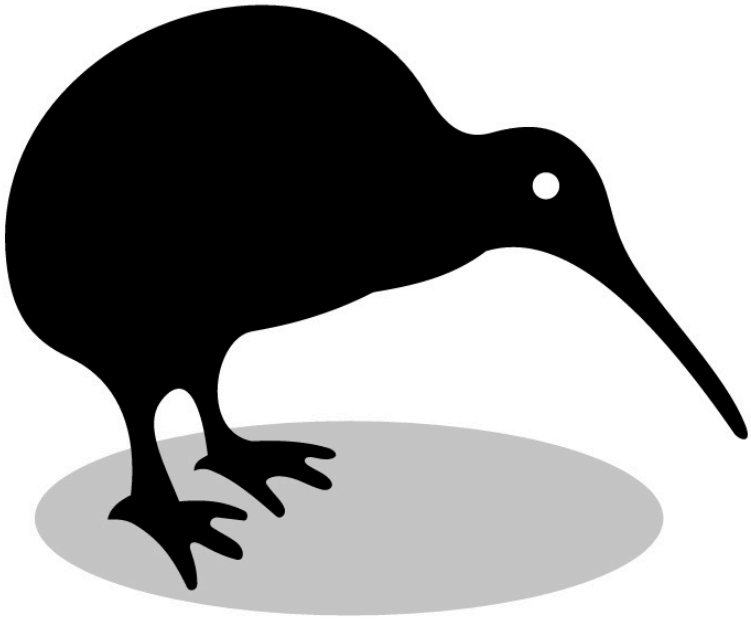


Figure 6-3. *This Adobe Illustrator vector-graphic drawing was saved in SVG format and placed in an HTML page with the <img> tag. (Kiwi drawing courtesy of Chris Coyier, CSS-tricks.com.)*

Working with Images

Compression. Depending upon the file format, bitmapped images can be compressed to varying degrees. Lossless compression reduces the file size while retaining image quality. Lossy compression, on the other hand, reduces file size further by discarding some information from the image, which lowers the quality.

Photoshop. If you have Photoshop, the best way to save a bitmapped image for the web is to use the File > Save for Web dialog box (**Figure 6-4**). It offers a choice of contone and indexed formats along with options for colour levels and compression.

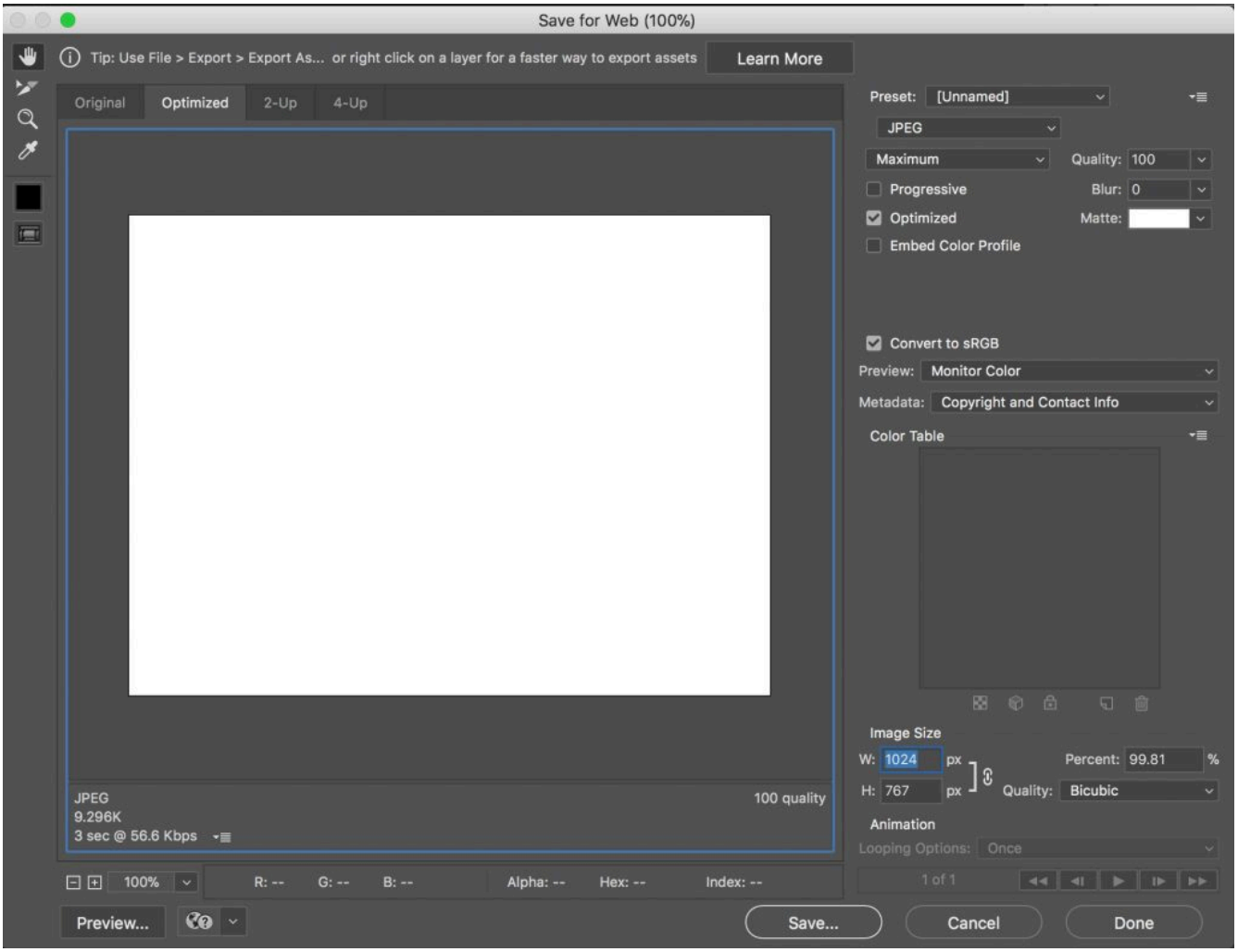


Figure 6-4. *Photoshop's Save for Web dialog box offers the opportunity to preview and save images in various file formats.*

Photoshop's Slice tool is a convenient way to divide an image into multiple zones, each of which can be saved in the most suitable file format. This would be convenient if you created a site mockup in Photoshop and wanted to use parts of the mockup in the actual web site.

Apple Preview. Apple Preview can also be used to resize and crop images. They can be saved in TIFF, JPEG, and PNG formats.

Transparency. Transparency means that if part of a bitmapped image is masked, or a vector graphic does not contain a filled object as its background, the web page's background image or colour will show through the image. Formats that support transparency include GIF, PNG-24, and SVG.

Chapter 7 - Video

Any YouTube fan can attest to the explosion of digital video on the web and how it has revolutionized communication—be it for education, product information, or just sharing personal experiences.

Most digital cameras today can record standard-definition (SD) and high-definition (HD) videos (**Table 7-1**). Camera types range from cell phones to point-and-shoot, advanced amateur, and single-lens reflex (DSLR) cameras, along with amateur pocket video cams and shoulder-mounted professional models. Higher-level cameras offer more professional features, such as more control over camera settings and stereo recording from an external microphone.

Table 7-1. Video Resolutions

Format	Resolution (pixels)	Suggested Use
Standard Definition (SD)	640 x 480	iPad, locations where space is limited
High Definition (HD 720)	1280 x 720	small-screen TVs and standard video window on laptops
High Definition (HD 1080)	1920 x 1080	large-screen TVs and full-screen on large monitors

Since digital video consists of a series of bitmapped images, file size limits their length and quality. A video is like a series of Photoshop images at 24–30 per second. If the length and width of the frame are doubled, then the file size quadruples. File size is a major consideration in the length and quality level of the video.

File format is another important consideration (**Table 7-2**). Choose a format depending upon whether your movie will be played on a Windows, Apple, or tablet computer.

Table 7-2. Video Formats

Format	File extension	Use
Apple QuickTime	.mov	MacOS and iOS devices
MP4	.m4v, mp4	MacOS, iOS, and Windows
Windows Movie Player	.wmv	Windows computers Apple computers with plugin

Video Programs

A video editing program enables you to edit your video by doing such things as:

- deleting scenes that you don't want
- splicing scenes you do want into a continuous sequence
- adding transitions between scenes
- adding titles
- creating special effects, such as time-lapse
- deleting sound and adding a voiceover (narration)

Video Tags

When coding in HTML5 the <video> tag can be used to insert a video. As with images, the height and width can be specified, along with whether the user has controls (start, pause, stop).

Example:

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

Apple iMovie

iMovie is an easy-to-use program for basic video editing, provides an appropriate example of a video editing program for a one-semester course in web design.

Opening a Movie

1. Transfer your unedited movie from your camera to your computer.
2. Open iMovie and select one of the readymade themes (or No Theme).
3. Select Import Media and open your unedited movie, which will come into the Clips window.
4. In iMovie, select the entire clip and move to the editing window.

Deleting Unwanted Footage

1. In the editing window, place the play head where you want to cut the clip. Placing a marker (M) is a convenient way to evaluate a starting and stopping point. Then select Split Clip (B).
2. After splitting the clip you can select and delete the sections you don't want.

Adding Transitions

1. From the Content Library, select Transitions.
2. Find the transition you want and drag it in between two clips, or at the end of a clip. In the example, a Fade to Black transition was added to the beginning and end of the movie. The length of the transition can be set in seconds and tenths.

Adding Titles

1. From the Content Library, select Titles.
2. Find the title you want and drag to the location where you want it to appear—at the beginning or end of the clip, or in the middle. None of these locations add time to the movie. The title can be placed before the start or after the end of the clip, which lengthens the movie.

Lowering or Deleting Sound and Adding Voiceover

1. To lower the volume in a clip, select the clip and, using the Volume Adjust control above the play window, reduce the volume 100% to the level you want. The volume can also be faded in and out by dragging the dot above the volume level line in the sound window.
2. To delete sound, select the clip and then choose Modify > Detach Audio. The audio will appear below the clip as a green line. Select this line and delete.
3. To add a voiceover, select the clip and move the playhead to the location where you want to start the narration.
4. Choose Window > Record Voiceover. The microphone icon will appear below the play window.
5. Click the start button, speak into the microphone, and click stop when finished. The narration will be added to the clip.

Exporting

1. To export your edited video, click the Share button (or File > Share), then select File.
2. The dialog box offers a choice of resolutions for output, along with an estimate of the file size. Note that file sizes are proportionally larger with HD-720 and HD-1080 than with SD.

Chapter 8 - Animation

The ability to show moving objects that respond to reader inputs has made vector graphic animation a valuable addition to the web (**Figure 8-1**). Think of vector animations as Adobe Illustrator drawings that move and interact with users.

Figure 8-1. This bouncing ball animated in Tumult Hype is a classic animation used to illustrate the stage, drawing tools, timeline, keyframes, and scripting through the use of control buttons.

Vector vs. Bitmapped Graphics

The value of vector graphics is that they are defined by mathematical formulas, not like bitmaps as in photos or Adobe Photoshop files. Therefore the files have smaller sizes and load more quickly into the browser.

To appreciate the difference between vector and bitmapped graphics, think of a green square that's 100 pixels on each side. A bitmap creates the square using 100×100, or 10,000, green-coloured pixels, each with 8-bit RGB colour, for a total of 16.7 million colours. If enlarged beyond 72-ppi screen resolution, the individual squares would be visible and appear pixelized, or stair-stepped. If enlarged two times at the same resolution, the number of pixels would quadruple to 40,000.

In a vector graphic, the square is defined by a mathematical equation. If enlarged, it appears smooth while the file remains the same size. Thus vector graphic animations are characterized by small file sizes and scalable dimensions that reproduce with good resolution.

Animated vector graphics were popularized by Adobe Flash, introduced in 1996. Flash saves files in the proprietary .fla format, which can be exported to opensource .swf for viewing in applications, in browsers, and with Adobe's free Flash Player utility. (SWF stands for Shockwave Flash, in reference to the program's original creator.)

With the introduction of the iPhone, iPad, and iOS operating system, Apple refused to support Flash, citing security flaws in the proprietary file format. Since then the use of Flash has declined, and more animations are being created in HTML5. Flash users who want to create HTML5 animations can export .fla files with Google's free Swiffy extension or use a dedicated HTML5 animator like Adobe's Edge, Tumult Hype, and a variety of other programs. Still, the animation concepts are the same.

Animation Concepts

Key requirements for an animation program include (**Figure 8-2**):

- *Stage.* The screen, or drawing board, or pasteboard, are represented as a "stage" for the moving objects.
- *Drawing.* Users must be able to draw objects, including points, lines, curves, fills, and strokes.
- *Timeline.* To make objects move on the stage at the desired speed, users must have a timeline, measured in frames/sec., and in the total number of frames. Commonly used frame rates are 20, 24, and 30 fps. Round numbers facilitate calculating the total number of frames needed for an animation of a specified time.
- *Keyframes.* A concept borrowed from film animation, keyframes identify transition points in the action. The program "fills in the blanks" in between user-drawn keyframes through the process of tweening.
- *Layers.* To organize and work easily with multiple moving objects, it is convenient to place each object

on a separate layer.

- *Scripting language.* A control language enables users to control and interact with the animation. Interactions include starting and stopping the animation, or moving to specified points in the timeline.

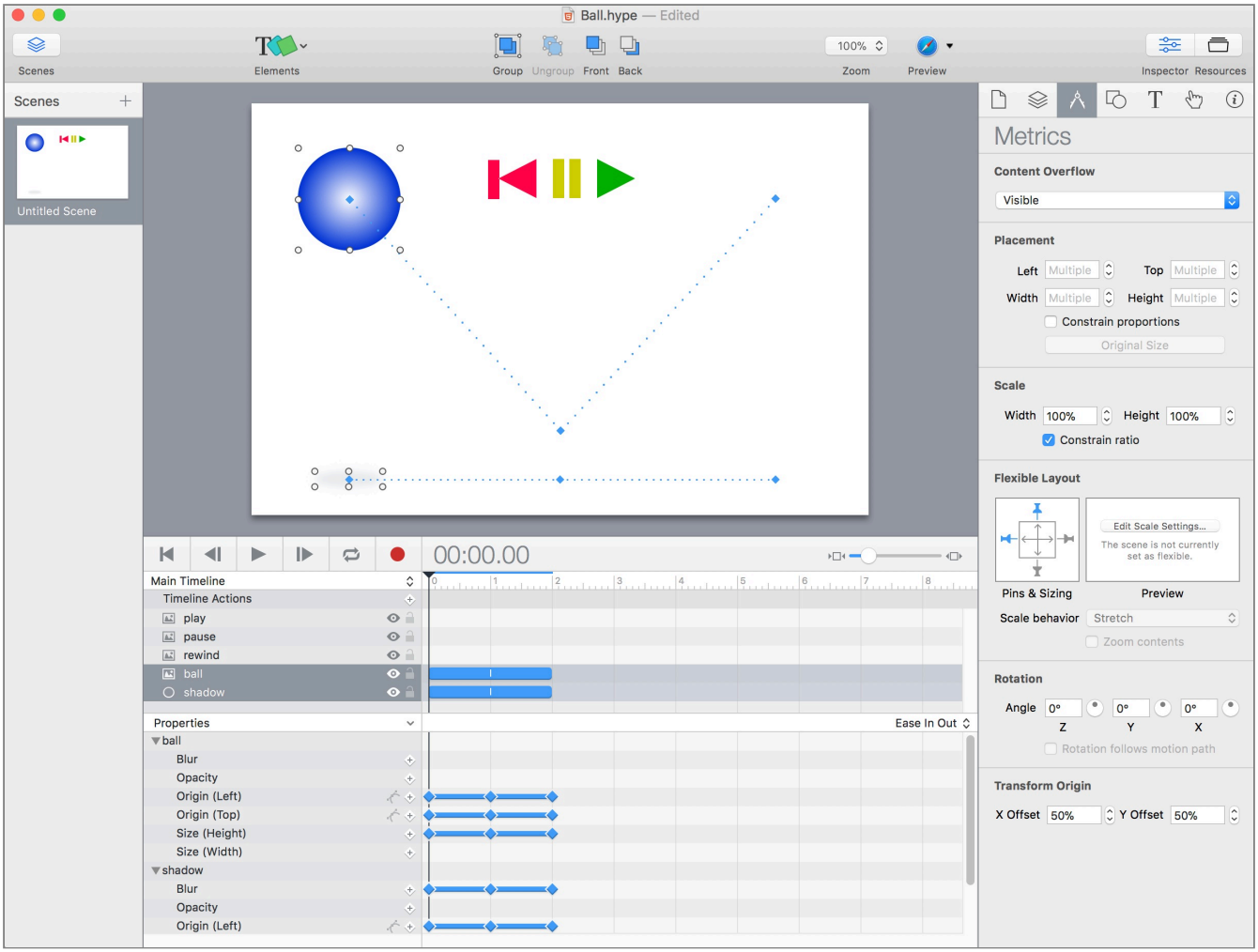


Figure 8-2. This screen capture of the ball animation in Hype shows the objects on the stage in separate layers, timeline in seconds, keyframes (blue diamonds), and "tweens" (blue lines) or intermediate frames.

Methods of Animation

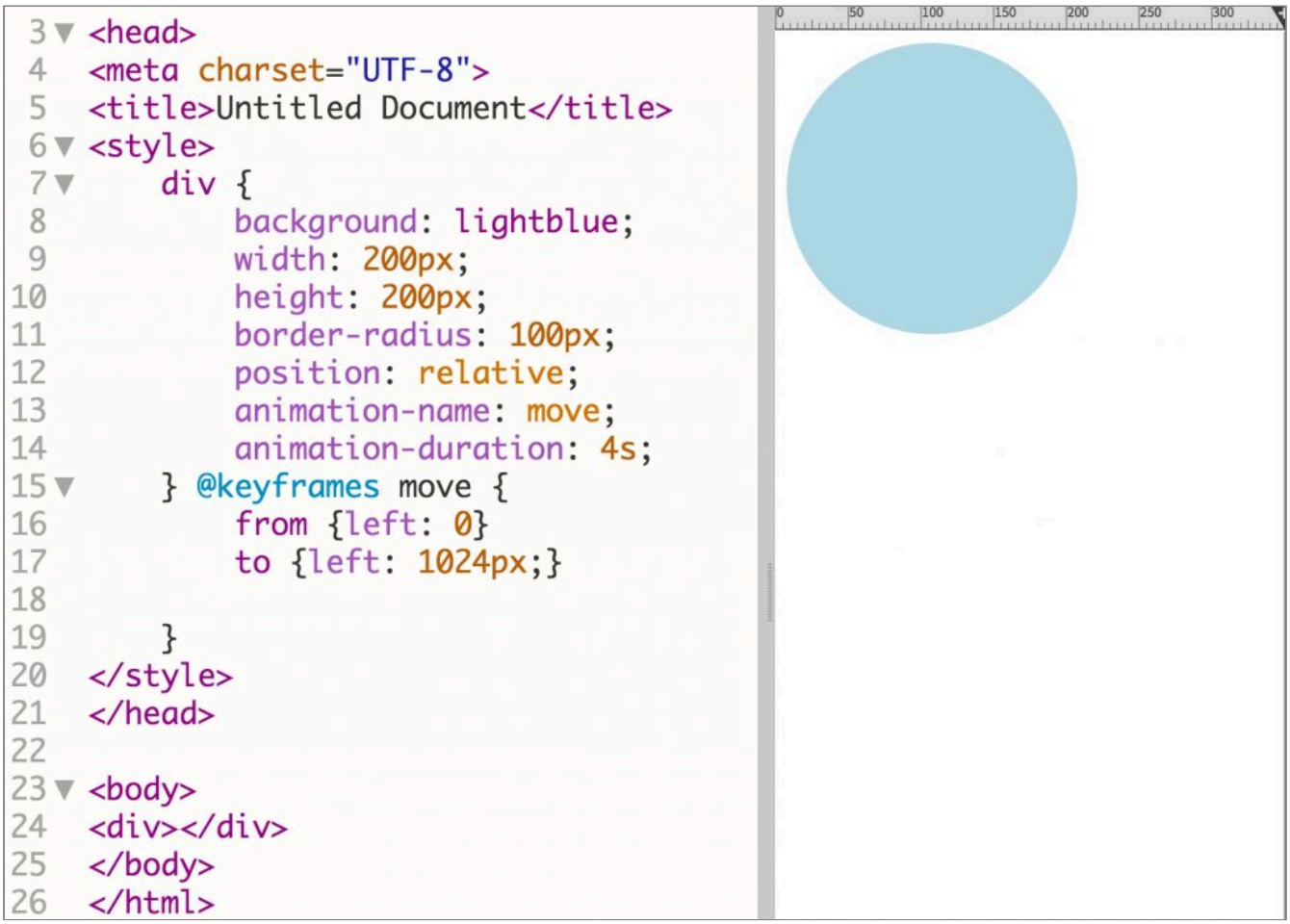
There are several ways to create animations for the web, including the CSS animate style, JavaScript and the jQuery library with its animate functions, and animation programs like Tumult Hype and Adobe Animate.

CSS Animate

HTML objects can be animated using the CSS @keyframes rule and animate style. A CSS animation changes a tag from one style to another at a specified interval. The changes (e.g., colour, position, size) must be specified

in the @keyframes rule inside the <style> statement. The @keyframes rule needs to be applied to an object using the animation-name and animation-duration styles.

Figure 8-3 shows a simple example of the @keyframes rule and animate style applied to a circular div.



```
 3 ▼ <head>
 4   <meta charset="UTF-8">
 5   <title>Untitled Document</title>
 6 ▼ <style>
 7 ▼     div {
 8           background: lightblue;
 9           width: 200px;
10           height: 200px;
11           border-radius: 100px;
12           position: relative;
13           animation-name: move;
14           animation-duration: 4s;
15 ▼      } @keyframes move {
16           from {left: 0}
17           to {left: 1024px;}
18
19       }
20   </style>
21   </head>
22
23 ▼ <body>
24   <div></div>
25   </body>
26   </html>
```

Figure 8-3. Example of the @keyframes rule (line 15) and animate style (lines 13 and 14) applied to a circular div. When the page is loaded, the div moves across the page from left to right.

Table 8-1. CSS Animate Styles

style	description
animation-name	name of @keyframes rule
animation-duration	time in seconds (s) to complete animation
animation-delay	time in seconds before animation starts
animation-iteration-count	number of times the animation plays (number or infinite)
animation-direction	normal, reverse (backwards), alternate (forward, backward), alternate-reverse (backward, forward)
animation-timing-function	linear, ease, ease-in, ease-out, ease-in-out
animation-fill-mode	none, forwards, backwards, both

User Interaction

A CSS animation can be triggered when the reader has the mouse over the object or when the user clicks on the object by applying the animation styles to the :hover or :active states. In the example shown in **Figure 8-3**, the blue circle could start to move if the animation styles in lines 13 and 14 are applied to the div:hover (mouse over) or div:active (mouse click) selector.

To start, stop, or pause the animation using a button, then you would need to use JavaScript or jQuery.

Animation with jQuery

The jQuery JavaScript library can be used to animated objects using the jQuery animate method. The animate method produces an animation by modifying one or more CSS styles, such as left or right position on the page.

Syntax: (selector).animate({styles},speed,easing,callback)

Where:

- selector is the ID or class of the item to animate
- styles is a list of styles to be animated
- speed is the duration of the animation, specified in milliseconds or as “fast” or “slow”
- easing is the relative speed at the beginning and end (linear or swing)

Figure 8-4 shows an example of a simple animation done with jQuery. When the “Go” button is pressed, the motorcycle moves 1200px to the right. When “Come Back” is pressed, it moves 1200px to the left.

```
25 <body>
26   <button id="go-button">Go!</button><br/>
27   <button id="comeback-button">Come Back!</button>
28   <img id="motorcycle" src="motorcycle.png" />
29
30   <script src="http://code.jquery.com/jquery-3.3.1.min.js">
     </script>
31   <script type="text/javascript">
32   $(function() {
33           $('#go-button').click(function() {
34               $("#motorcycle").animate({left:
                 '+=1200px'},1000);
35           });
36
37           $('#comeback-button').click(function() {
38               $("#motorcycle").animate({left: '-
                 =1200px'},3000);
39           });
40           });
41   </script>
```

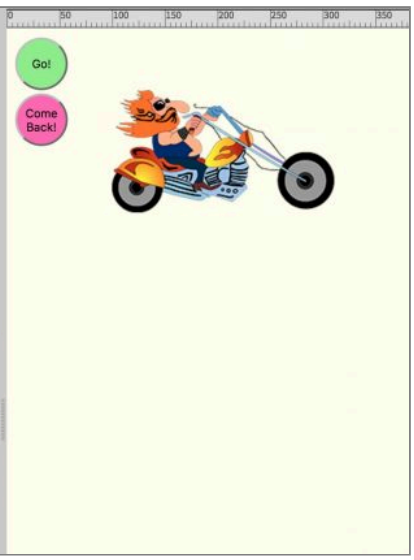


Figure 8-4. Simple animation done with jQuery, which allows for user interaction via the two buttons.

Readers are referred to Chapter 10 JavaScript Libraries and to the jQuery site for more details.

Animation with Tumult Hype

Tumult Hype (**Figure 8-5**) is an easy-to-use program that creates animations in HTML5. The animations can be exported as HTML for placement in web pages and as Apple Widgets (.wdgt) for direct placement in Apple iBooks Author.

Similar to Adobe Flash, Hype uses the timeline and keyframe models to create animations. Buttons can be added to start, stop, pause, and continue the animations.

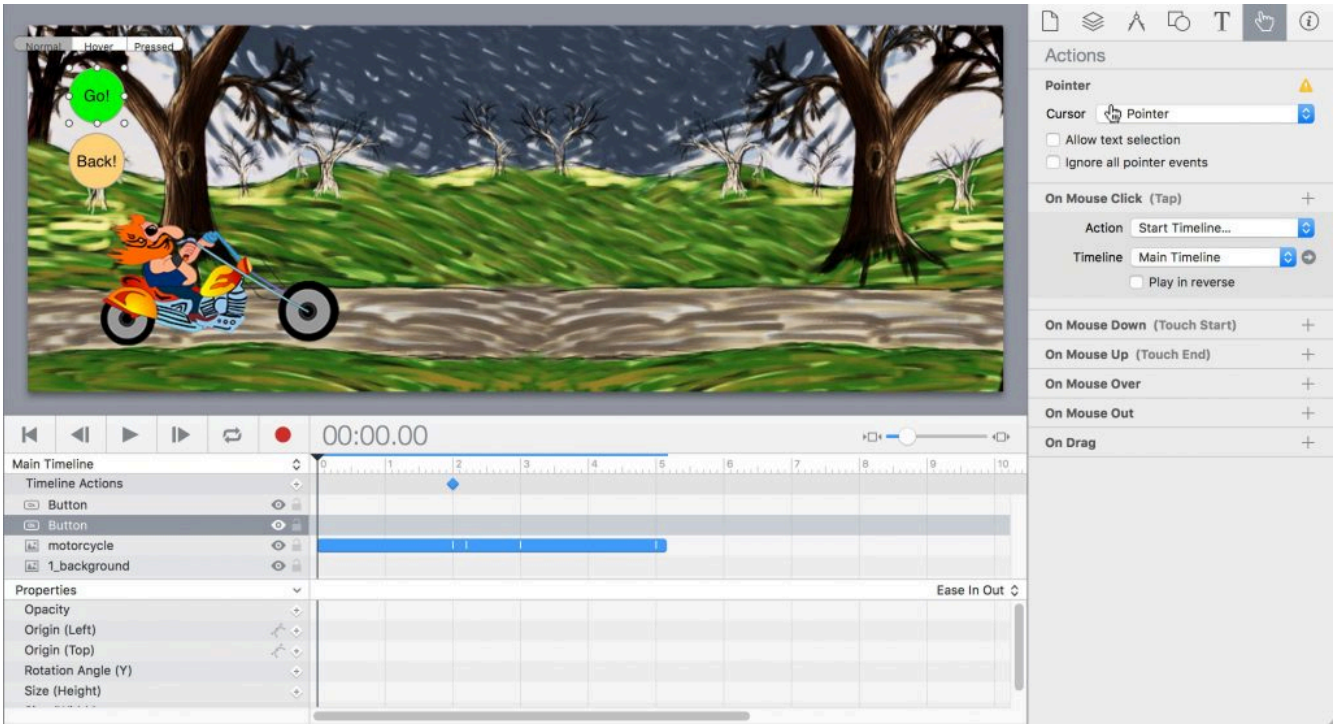


Figure 8-5. Screenshot of Tumult Hype animation showing the stage, timeline, keyframe properties, and inspector.

Chapter 9 – JavaScript

The primary role of JavaScript is to modify the default behaviors of a web browser. Browsers come programmed with standard approaches to how a web page is rendered and interacted with. Therefore, if you wanted to add or modify an interaction, you'd need to leverage JavaScript to make this happen. Here are some examples of what can be achieved with JavaScript:

- Show a *spinner* while the page loads (known as a pre-loader). The browser is made to overlay an image while the page renders. Once rendering is completed, the overlay is hidden and the content beneath is revealed.
- Changing how a form submits data. By overriding a form's *action*, the standard HTML submission is handled by JavaScript which will submit the form asynchronously (without reloading the page).
- Tracking a user as they navigate your website. JavaScript is used by analytics companies to allow page requests and interactions to be sent to a third-party website. Here are a few analytics companies that provide analytics: Google Analytics, KissMetrics, and Piwik.
- Having an interaction in one area of your website cause a response. Image sliders and carousels are examples of this. Click on a next or previous button to load a new image.

It's easy to start coding your first script. All modern web browsers ship with JavaScript, so there are no external assets required to program using *Vanilla* or pure JavaScript. Most developers prefer to use a JavaScript library because they ship with many functions to simplify code. Other developers feel external libraries add a lot of unnecessary bloat since most websites use a fraction of the code available. In either case, JavaScript on its own is powerful enough to cover most use cases.

Pro-tip: Don't mix JavaScript up with Java. JavaScript is a web language while Java is mainly used to create applications and control embedded circuits. Until recently, JavaScript was only used for frontend or browser-based scripts. Advances in web technologies now allow it to be used on the backend using a language such as *Node.js*. This means you only need to know JavaScript to control browser behaviours *and* load content from a database and pre-process content before transferring it to the user's computer.

Writing Your First Script

The simplest starting point is making an alert appear in the user's browser window. In the example below, an inline *onclick* attribute is used to run a JavaScript alert.

```
<html>
<body>
  <h1>Your First Script</h1>
  <button onclick="alert('Hello World!')">Click Me</button>
</body>
</html>
```

Figure 9-1 reveals what you would see when you click on the button:

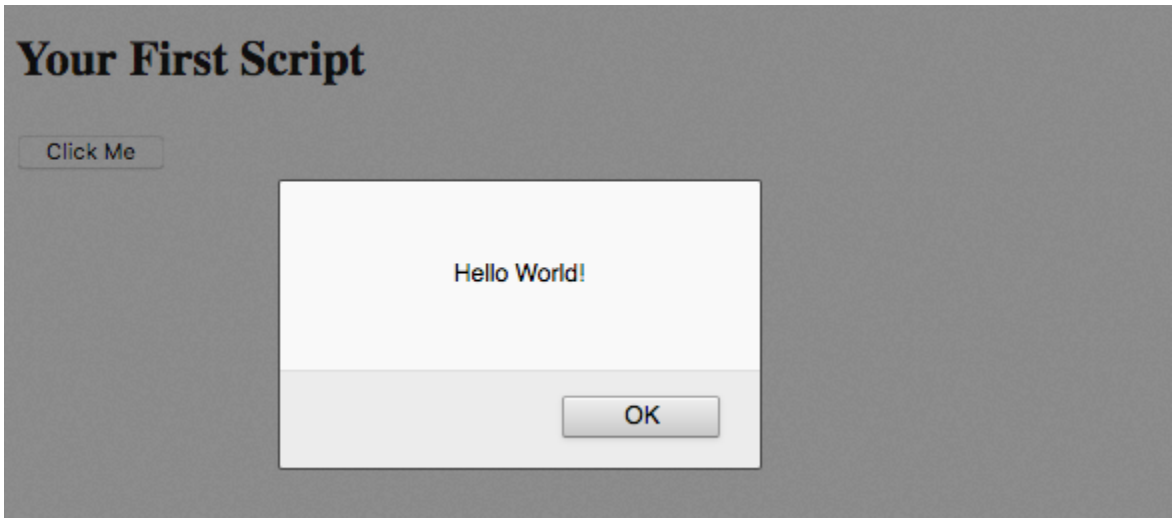


Figure 9-1. Javascript Alert

Capturing and Outputting Content

In this example, we will be taking content from an *input* field and outputting it in a *p* (paragraph) tag. For this to work, we will need to use the *id* attribute so that we can target the elements.

```
<input id="sometext" value="enter anything">
<button type="button" onclick="outputText()">Click</button>
<p id="output"></p>

<script>
function outputText() {
  document.getElementById('output').innerHTML = document.getElementById('sometext').
```

```
}
</script>
```

This time, we'd separated the script from our HTML. In the previous example, JavaScript was kept inline using the *onclick* attribute. By separating the JavaScript and keeping it before the closing body tag (hidden in above example), we make our code more manageable and less polluted. The *onclick* attribute called the JavaScript function called *outputText*.

Can you see the relationship between the *id* attributes and the script? We are setting the HTML within *output* equal to the value of the tag *sometext* using their *id* attributes for identification. The standard is to only use an *id* only once on each page. An ID such as a driver's license is made to be unique to its holder. Duplicating the same ID is counterfeiting, so avoid doing it with your code.

How JavaScript Works: the DOM

In order to make changes to the browser and the site's content, JavaScript needs access to the model used to store the browser and page's parameters. Each parameter is stored as a node in a tree or database called the *Document Object Model*. Each branch of the document's data is known as an object. This allows for *traversing* which just means that JavaScript is tracking and accessing all tags in your HTML file. In the example of on *Capturing and Outputting* content, our document has several *id* attributes. JavaScript searches our HTML elements for a tag with the *id* specified.

Let's break down this code, *document.getElementById('output').innerHTML*. Each object is separated by a period (.). The first object is the *document* and we're looking within it for an element with an id of *output*. The function in JavaScript called *getElementById* is what we use to traverse the document and returns it as an object. The object is composed of several attributes, one of which is the *innerHTML*. For an *input* element, we use the *value* attribute to access the input field's current value. If the user types something into the input field, JavaScript will capture it.

Another example of an object that we can access is the *window*. The window contains details on the browser's window. We can use this to capture the user's location as *x* and *y* coordinates.

Mouse Location Example

In the following example, the *onmousemove* attribute is applied to the *html* tag. Using this method allows the function to only run when the mouse is moving anywhere on the page. If applied to the *body* tag, the result would be only executed on the visible contents of the page. We could have used *onclick* instead, but you'll find real-time updates quite satisfying compared to clicking to have the value update.

The *event* object is passed into the *showXY* function. The object provides the user's X and Y mouse position so that we can output it into the element with an *id* of *location.* In this case, it's a *span* element. The function sets the location span equal to *event.clientX + "," + event.clientY*. Note the use of a plus (+) sign to append a comma (,) to the value of *event.clientX*. A string should be encapsulated with quotes. It does not matter if it's a double quote

or a single quote. However, if you want to use double quotes within your text, you'll need to encapsulate it with single quotes (and vice versa). For example, x = "There's a small image" and 'They said, "upload the image" and clicked.'

Figure 9-2. Mouse XY position tracking in Mozilla Firefox (v.58) using JavaScript code provided below

```
<!DOCTYPE html>
<html onmousemove="showXY(event)">
<body>
<p>XY
  (<span id="location">0,0</span>)
</p>
<script>
  function showXY(event) {
    var coords = event.clientX + "," + event.clientY; // Setup a variable called coo
    document.getElementById("location").innerHTML = coords; // Output the value in #
  }
</script>
</body>
</html>
```

It's always a good habit to provide placeholder content. In the location span, *0,0* is provided. This content is replaced once the current mouse position is captured by the browser and processed using the script. In cases where a user is on a slow connection or waiting for external scripts to load, the placeholder content will greet them and provide some direction.

Chapter 10 – JavaScript Libraries

A JavaScript library is a collection of function that speed up the development of a web site. There are a range of libraries available and many have strong communities that assist in improving the functionality and supporting users. This chapter provides an overview of popular libraries and how you can get started with them.

When choosing a library, it is important to assess your application requirements and experience. Some libraries lack documentation, especially if newly released. Mature libraries have an advantage when it comes to community support, documentation, and life-cycle. Younger libraries may be short-lived or require code to be rewritten when new versions are released. However, the benefit of a younger library is that they often employ novel approaches and may be more efficient in execution. Also check for corporate backing when looking at a library. This ensures the code will continue to be supported and updated.

In this chapter, we will look at jQuery (mature) and Vue.js (emerging/young).

For the purpose of this book, a JavaScript *library* is a single file that can be included with your code and provides several functions for your use. This differs from a framework, which is more opinionated on how you develop your entire site and relies on multiple files. Frameworks often provide both frontend and backend code. A JavaScript *plugin* extends a library to provide added functionality. Plugins rely on a library, whereas libraries are self-sufficient.

List of JavaScript Frameworks

- Ember
- Angular (Google)
- React (Facebook)

List of JavaScript Libraries

- jQuery
- Vue.js
- Prototype

- MooTools

Getting Started with a JavaScript Library

The standard is usually to download the library and move the files into your development folder. This process makes sense when you're developing without an internet connection or if you're creating a standalone website for an internal network. What makes the web so powerful is its inter-connectivity. So why not leverage this? Why don't we just link to a host that already has the file so we have less to manage in our files and folders. But don't just use any host, use a *Content Delivery Network*.

Content Delivery Networks (CDN)

CDN's are servers that are typically used to host assets so that sites aren't bogged down serving them. The advantage is that they are located all over the world, so if a user connects to your website, they can be served assets (images, videos, static code) from the server closest to them. We still need our own server for anything that's dynamic, meaning anything that changes often. If you're running a database, you can package what the user needs, while have them download assets elsewhere. The result is a quicker load time.

Another advantage of a CDN is caching. Let's say your user visits a WordPress blog and is given the jQuery file from a CDN called *Cloudflare*. Then your user heads over to your site and you've linked to the same jQuery file on *Cloudflare*. The browser has already downloaded it, so there's no point in downloading it again, it will just use the version it already has. CDNs typically store each version of a file in a separate folder so it will never expire. Your browser caches files for a set period of time before it downloads again. The expiry date can be specified by the CDN to be a very long time.

We'll be showing you how to use a CDN to use jQuery and Vue.js

jQuery

Released in 2006, jQuery is the most popular JavaScript Library online. Thousands of plugins exist and their documentation and community support is extensive.

To start, you can find the latest version of jQuery on a cdn.js. Simply find the *min* (minified) version and copy the *script tag*.

Figure 10-1. A screenshot of the jQuery library on cdnjs.com

The code you've copied should be placed in your HTML document before the closing body tag:

```
  <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.3.1/jquery.min.js"></
</body>
</html>
```

Once that's completed, you're ready to write your first line of jQuery code!

Here's an example of a button that shows and hides all paragraphs in a document:

```
<!DOCTYPE html>
<html>
<head><title>Button Toggle</title></head>
<body>
```

```
  <button>Click on me</button>
  <p>This text will appear and disappear when you click the button</p>

  <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.3.1/jquery.min.js"></
  <script type="text/javascript">
    $('button').click(
      function() {
        $('p').toggle();
      }
    );
  </script>

</body>
</html>
```

One of the benefits of jQuery is its use of CSS-like targeting to control your interactions. To target an element, we just use its name, in this example, it's *button*. We then specify the interaction as a *click*. The function that follows runs when the *button* is *click*ed and targets the all paragraph elements on the page to *toggle*. By switching out *toggle* for *slideToggle*, we can animate the paragraphs to vertically slide in and out of view.

Try changing the *toggle* to *slideToggle* and adding more paragraphs to the document. Then load the page in a browser and click on the button to see jQuery's magic in action. Another function to try is *fadeToggle*.

jQuery Counter

To create a counter, we will need to use a variable to hold the current count. The counter will be set to zero by default. Each time a button is pressed, it will increment by one and the value will be output in HTML. To do this, we need to specify the output location and the button. Instead of using HTML elements (*button, p* or *a*), we will use *id* attributes.

For counting down, instead of using the jQuery *click* function, the *onclick* attribute will be used on the *button* element. A function called *subtract()* will need to defined in JavaScript.

```
<button id="add">+</button>
<button onclick="subtract()">-</button>
<p id="output">Number appears here</p>

<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.3.1/jquery.min.js"></sc
<script type="text/javascript">
 // Setup the variable and output it
 counter = 0;
```

```
 $('#output').html(counter);

 // Adds 1 when the button with id='add' is clicked
 $('#add').click(
   function() {
     counter++; // same as counter = counter + 1;
     $('#output').html(counter);
   }
 );
 // Subtracts 1 when the button with the onclick="subtract()" is clicked
 function subtract() {
   counter--; // same as counter = counter - 1;
   $('#output').html(counter);
 }
</script>
```

Vue.js

As an up-and-coming library, Vue.js has excelled due to its ease-of-use and efficiency. The syntax is cleaner compared to jQuery, but you'll be polluting your HTML with new attributes specific to Vue. This means the separation of HTML and JavaScript is less apparent. The benefit is how fast Vue is at updating the Document Object Model (DOM). Think of the DOM as the HTML Lego blocks that are used to construct your website. Each block is accessed and acted upon by JavaScript. If you have a lot of code and lots of content updates, the browser will be working harder to traverse (scan your code) and make relevant updates. Vue has a very efficient means of performing updates compared to jQuery.

Just like jQuery, including Vue.js is as simple as a *script* element link from a CDN.

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/vue/2.5.13/vue.min.js"></script>
```

Hello Name Example

```
<div id="example">
  <input type="text" placeholder="Enter your name" v-model="username">
  <h1>Hello <span v-html="username">Sample Name</span>!</h1>
</div>

<script src="https://cdnjs.cloudflare.com/ajax/libs/vue/2.5.13/vue.min.js"></script>
<script type="text/javascript">
new Vue({
  el: '#example',
```

```
  data: { username: '' }
})
</script>
```

In the above example, we must first define a *div* to run our example. All HTML that will be acted upon will be nested within the *div*. The *input* element as an attribute of *v-model*, which is a Vue.js attribute that allows us to link the input content to the *span* element with an attribute of *v-html*. Notice that both have a value of *username*, allowing the *input* and the *span* to relate to each other.

In JavaScript, we've specified the location where Vue.js will be focusing its efforts *(#example)*. By default, the username is set to nothing. Notice how the "Sample Name" is removed when the page loads. We override the content when Vue.js loads our data.

Vue.js Counter

```
<div id="counter">
  <button v-on:click="number++">+</button>
  <button v-on:click="number--">-</button>
  <p v-html="number">Number appears here</p>
</div>

<script src="https://cdnjs.cloudflare.com/ajax/libs/vue/2.5.13/vue.min.js"></script>
<script type="text/javascript">
new Vue({
  el: '#counter',
  data: { number: 0 }
})
</script>
```

There is no need to explicitly output the number. Vue.js handles this because it's directly linked to the *v-html* attribute. We've also skipped the need to define a function because we can run an arithmetic process directly on a *click* action. For more complex operations, a function can be called using *v-on:click="functionName"*. We've done away with about 10 lines of code using Vue.js compared to jQuery.

Chapter 11 - Uploading Content to a Web Server

To place pages on the World Wide Web, you will need (1) a domain name and URL and (2) a web server with sufficient space for your files (pages, images, video, audio, and so on).

Domain names are issued by designated sites such as GoDaddy.com, Domain.com, and numerous others. These sites have a search function where prospective users can see if their desired URL is available. Domain names rent usually annually for $10 and up. Popular names may cost much more.

The domain provider will offer a "pointer" (**Figure 11-1**) to connect your URL to the server where the content is located.

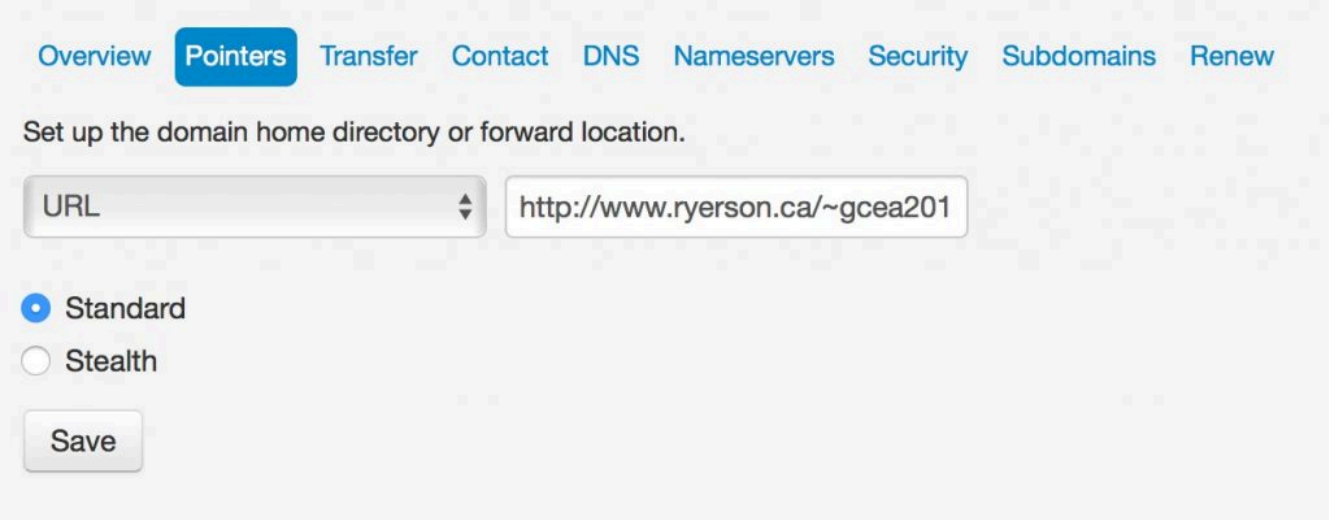


Figure 11-1. The "pointer" setting on Domain.com brings readers who type the URL into their browsers to the actual server location.

If you're studying web design in a university, high school, or continuing education class, the institution may provide a web server and URLs for students to use during the class.

If you have a URL and service space, you will need a file transfer protocol (FTP) program to upload files to the server and set reader permissions. A multitude of free FTP programs are available for download. At Ryerson University we use FileZilla to place files on a student web server maintained by the Computing and Communication Services department.

File Preparation and Upload

To upload files to a web server, the authors recommend the following steps:

1. Make a “gold master” folder on your desktop, such as “site,” and place final files in it.
2. Place the required .html, .css, and image files for your site into the folder.
3. Follow naming conventions for convenience:
    - Use all lowercase letters.
    - No spaces in file or folder names — use hyphen (-) or underscore (_) to separate words.
    - No absolute links to images, .css, .js, or other files (e.g., “file///MacintoshHD/images/image.jpg”) in your documents.
4. Make sure your home page is called “index.html” so the file will load automatically when the folder is entered.
5. If placing image, .css, and other files in folders, be sure to cite the folder name in the link. E.g., “images/image1.jpg”
6. If placing .html files in a folder, remember you will have to go up one directory level to get to the index file and other folders. Use “../” before the folder name.
7. Test the files to make sure all links work.

Avoid renaming any files after organizing the folder. The result is to break any links that have been coded since they no longer match to the filename.

Chapter 12 - eBook Formats

eBooks are electronic documents designed to be read on a tablet, laptop, or desktop computers. What distinguishes eBooks from other books is their ability to include media, like videos and animations, and their user-adjustable text size and reflowable format, which both facilitate reading.

Since they are easy to create, edit, and distribute, eBooks lend themselves to “professor publishing,” the writing of customized course packets for students. Students might also find them to be advantageous since they easily accessible, low cost, and portable (e.g., being able to carry multiple e-books on a tablet without the added weight).

The main disadvantage of eBooks is that the generic formats were developed for text-heavy documents like novels and business or psychology books that do not contain a lot of illustrations. These formats are not good for placement of multiple, related illustrations (e.g., a photo and accompanying diagram or graph) that need to be grouped together to illustrate a point or concept. Some grouping of photos is possible, however, this type of document could best be output as a PDF.

Interactive PDF

Interactive PDF is an Adobe Portable Document Format (PDF) file that’s intended for on-screen viewing. This type of PDF can have interactive features, including buttons, videos, music, animation, and links (**Figure 12-1**).

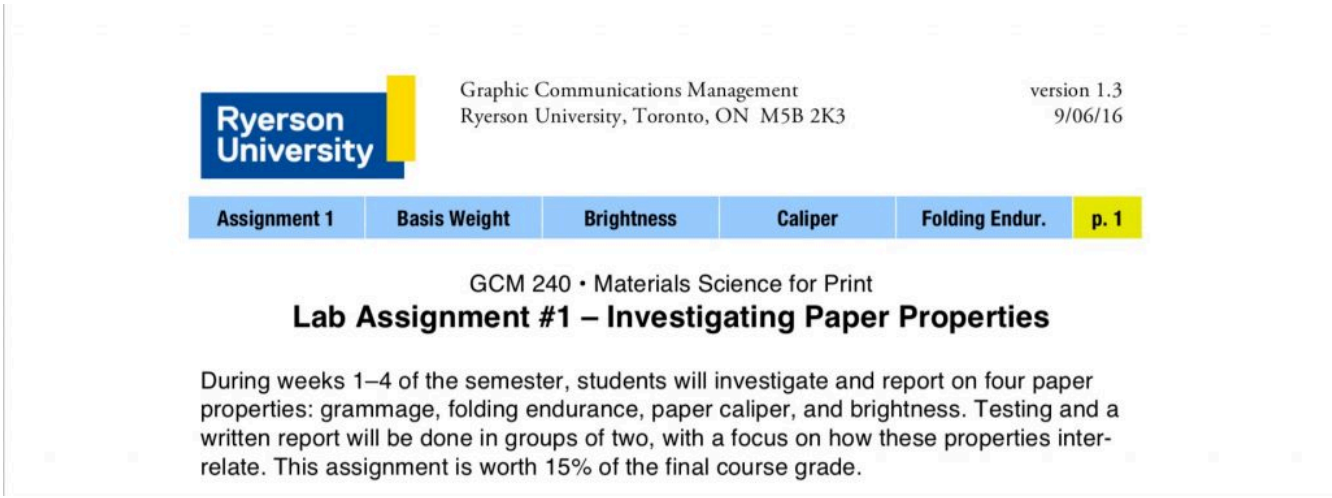


Figure 12-1. An interactive PDF class handout distributed via a distance learning program contains tabs at the top (blue background) that link to key pages.

Navigation buttons, for example, can be programmed to enable the reader to move from one page to the next and back, as well as the first and last pages, the table of contents page, and so on. If a table of contents is included, it can be interactive so that when the reader clicks on a section, the page will jump to that section.

The document could also include links to web pages that will load in a browser if the computer or tablet is connected to the Internet.

Among electronic or eBook documents, PDF is best suited to documents where the layout is most important, especially groups of related photos and/or diagrams with captions. For reading on tablets where text legibility is paramount, the eBook formats, ePub or iBook, may be a better choice.

What's Different about Interactive PDF? When designing an interactive PDF document, the designer should keep in mind what interactive features will be included and allow space for them. E.g., if navigation buttons are to be included, leave space at the bottom, top, or sides of the pages for the buttons. Placing the buttons on a master page enables them to be duplicated and positioned with accuracy on all dependent pages.

Pro tip: if you want to include an interactive table of contents, be sure to set up the contents using the Table of Contents feature and check "Create PDF Bookmarks" in the dialog box so the table of contents will work properly.

ePub

ePub (variously known as EPUB, ePub, .epub, and other abbreviations for "electronic publication"), ePub is a generic and open-source eBook format standardized by the International Digital Publishing Forum. ePubs can be created with InDesign and other page layout programs, including Apple Pages.

The advantage of ePubs is their reflowable text, which allows the reader to keep the text the same size, regardless of the eBook reader and orientation used. ePubs are best suited to text-heavy documents, such as novels,

psychology books, and business books. They can contain photos and graphics, but the reflowable text can present problems in formatting and layout.

ePubs are actually encapsulated web pages that are written with HTML5 and CSS3 code. Encapsulated means they are self-contained so that the reader does not need to be connected to the Internet to read the eBook. They can be opened and edited with Google's free Sigil ePub editor. Designers need to have a knowledge of HTML5 tags (e.g., <p> for paragraph, <h1> for heading one, and many others) and CSS3 to edit the code effectively.

What's Different about ePub? ePubs can be designed and exported from InDesign. When designing an ePub document, the page size is irrelevant, because pages will be created dynamically by the eBook reader. It may be most convenient to use a page size similar to the target eBook reader, such as an Apple iPad or Amazon Kindle.

Some design differences when creating ePubs (**Figure 12-2**):

- The "KISS" principle (keep it super simple) works best for ePubs — one text box, one photo/diagram, and one caption per page.
- All page items (text, photos, illustrations, etc.) must be inline in a single text box. Inline means the objects are pasted into the text box using the Text tool.
- All text needs to be styled, such as Body, Header, Caption, etc., especially when the Table of Contents Styles feature will be used.
- Don't build a table of contents because the eBook reader will make it. However, do make a Table of Contents Style because the export dialog box needs this to create the table of contents file for the ePub.
- Don't worry about automatic page numbers because pages will be numbered by the eBook reader app.
- Save the cover as a JPEG image file and added to the ePub in the export dialog box.

Figure 1. This example of a Sianxi papercut, a handmade watercolour, shows a finch with colourful flowers and foliage.

Origin

Paper-cutting originated from ancient activities of worshipping ancestors and gods, and is a traditional Chinese culture. According to archaeological records, it originated from 6th century,

Figure 12-2. A sample ePub file laid out in Adobe InDesign, following the 6-point "KISS" principle. Separate paragraph styles were created for the image, caption, heading, and text. All items are "inline" in one text box.

iBook

iBooks (file type .ibooks) are Apple's proprietary eBooks for the iPad. Currently, they can only be made with Apple's free iBooks Author program for MacOS computers. That is, you cannot lay out an iBook on an iPad or on a Windows PC.

Like ePubs, iBooks are actually encapsulated web pages based on HTML5 and CSS3. While restricted to Apple devices, they can contain a multitude of interactive objects, or "widgets" (**Figure 12-3**), including:

- photos and diagrams
- image galleries with multiple, related photos
- 3D objects in Google Collada (.dae) format
- 360° rotatable views of objects
- panoramic views of scenery that stretches across multiple photos
- video and audio
- encapsulated web pages and sites

The more complicated widgets need to be created in the companion program, iAd Producer.

Figure 12-3. An iBook laid out in Apple iBooks Author, showing examples of figures placed in specific locations.

What's Different about iBooks? iBooks Author has many of the same features as InDesign for designing iBooks: character and paragraph styles, table of contents, templates and master pages, tables, and guides.

Most items (including photos, widgets, and folders of multiple photos for creating an image gallery) are inserted into iBooks Author pages by drag-and-drop.

When finished, the iBooks Author file (file type .iba) can be exported to iBooks (.ibooks) and opened in iBooks for MacOS/iOS.

WordPress Pressbooks

Pressbooks is a relatively new web-based program for writing eBooks that was introduced by the web content management company WordPress. Pressbooks can be created online at pressbooks.com, then output as .epub and in .mobi format for Amazon's Kindle tablets. Kindle is Amazon's inexpensive and easy-to-carry line of eReaders that don't have the same functionality as tablets.

Questions to Consider when Writing an eBook for Class

An example of an eBook written for a class is provided by a web design class taught at Ryerson University. The class required a 400-page textbook that weighed 2 pounds (0.8 kg) and cost around $100. The book contained extensive information that was not used in the class. To use in place of this book, the instructors wrote a 45-page iBook that students could get for free from the class's distance learning site and read on lab iMacs or personal iPads.

Some questions to consider when deciding whether an eBook would be appropriate for a class:

Audience

- Will the audience be using primarily an iPad tablet, Android tablet, eReader, Mac computer, or Windows computer to read the book? Keep in mind that only iPads and Mac computers can read iBooks.
- Will the book consist mostly of text or also include photos, videos, animations, and other media? Media works best in iBooks.

Subject

"Yes" answers to any of these questions would favor the authoring of an eBook for class:

- Are you teaching a technology that's relatively new and for which there are relatively few textbooks?
- Are existing textbooks expensive and/or heavy or bulky?
- Do existing textbooks include a lot of information that's not relevant to your class?

Distribution

- Do you use a distance learning program to distribute information for your class? Keep in mind that eBooks—especially those with many photos and media—can be rather large files.

Chapter 13 - eBook Production

This chapter discusses different methods for producing eBooks. Let's assume as an example that the reader wants to write an eBook, "Introduction to Graphic Arts," of which one chapter will be devoted to large-format inkjet printing.

In preparation for laying out the eBook, the author could write the chapters in Microsoft Word, Apple Pages, Google Docs, or similar program. Let's assume all photos, videos, animations, and other media have already been created.

Apple iBooks Author

1. To lay out the chapter in iBooks Author (**Figure 13-1**), open a new document and select Import > Chapter from Word or Pages.
2. Drag-and-drop photos into the appropriate places on pages. The photos will automatically create a photo "widget," or interactive element, which can include a sequential number, title, and caption, along with a background and border. Photos can be anchored to specific locations or to text.
3. Drag-and-drop the video into the page. iBooks Author will optimize the video and create a widget as it would for a photo.
4. Let's say that the author wants to include an animation of the inkjet printing mechanism. The easiest way to include this in the iBook would be to do the animation in Tumult Hype, which can export animations as widgets.
5. To export the iBooks Author file to an .ibooks file, select File > Export > iBook.

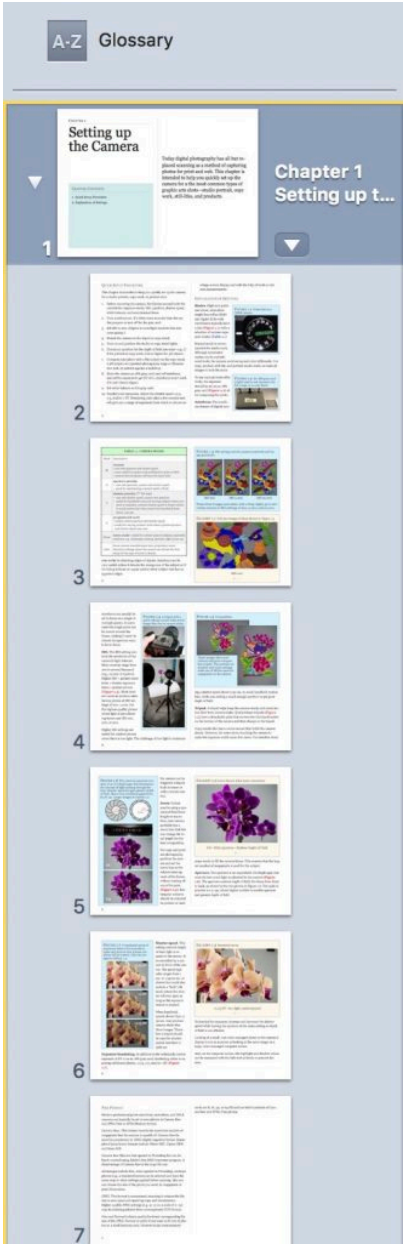


Figure 13-1. Sample iBooks Author chapter imported from a Microsoft Word document. iBooks are easy to create with the free iBooks Author program but are limited to reading on MacOS computers and iOS devices. However, they can be output as PDF for Windows and Android users.

Adobe InDesign

1. To create an ePub in InDesign, the authors recommend starting with a new document for Electronic Publishing at half the resolution of an iPad (512×386 pixels). The page layouts in InDesign (**Figure 13-2**) will look approximately as they will in the finished ePub.

2. Again, the chapter on Large-Format Printing (see above) can be imported into InDesign.
3. All page elements for ePubs need to be laid out “inline,” which means the page can only contain one text box (e.g., not separate text boxes for the body, picture captions, and sidebars). “Inline” means that photos need to be placed outside the page, then cut and placed with the i-beam tool as text.
4. Keep in mind that ePubs were designed to facilitate reading of text-heavy documents, not for complex page layouts with multiple, related graphics and captions. In other words, each page will look best if it includes only one graphic and related caption.
5. Videos can be placed but will only show up when the ePub is output as fixed-layout (FXL). Likewise, an animation would need to be saved as an animated GIF and therefore could not include interactivity (i.e., no start/stop buttons).
6. All text should be specified using Paragraph Styles in InDesign. Appropriate styles may include Body Text, Heading Text, Caption Text, and Figures (since inline photos behave as text).
7. If an automated Table of Contents Style is set up and specified upon output, it will create a table of contents that can be accessed on the eBook reader.
8. The cover should be created as a separate document, saved as a graphic (e.g., JPEG, PNG, GIF) and identified upon output.
9. To output the ePub from InDesign, select Export > ePub (Reflowable).

Figure 13-2. Sample ePub laid out in Adobe InDesign, with all items, including the figure, inline (behaving as text). The InDesign document for ePub needs to be set up with certain specifics, including that all items must be "inline" in a single text box, one illustration and caption (if any) per page, automatic table of contents, and cover created separately as a graphic.

WordPress Pressbooks

1. To lay out the Pressbook, go to Pressbooks.com and set up a new account.
2. Open a new document and place the text file containing the chapter.
3. Media can be added similar to iBooks Author and InDesign (**Figure 13-3**).

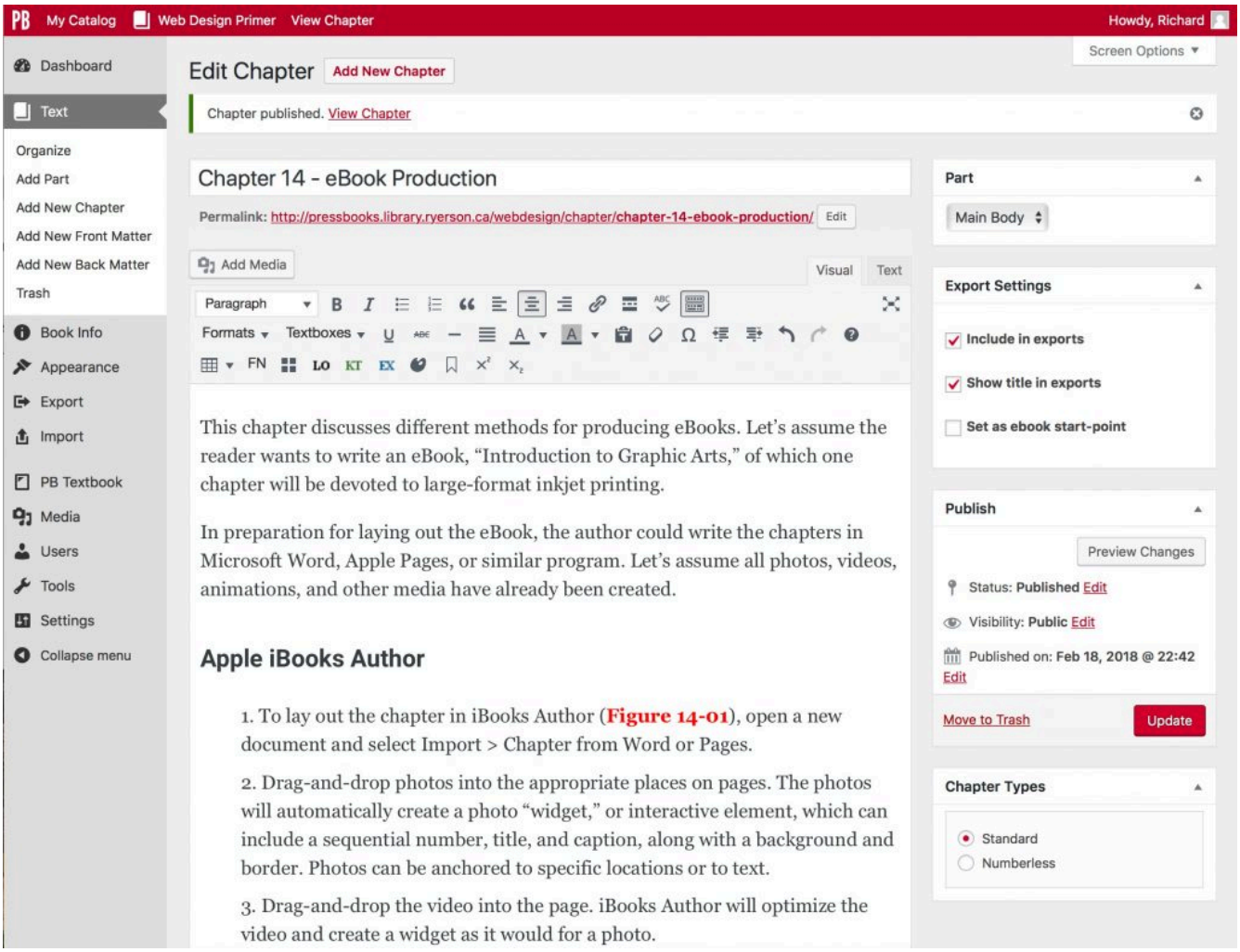


Figure 13-3. WordPress Pressbooks is an online authoring platform that works similar to WordPress's content management system for creating web pages.

Chapter 14 - Digital Photography for Web and Cross-Media

Since cell phones became popular, many web designers may have never worked with a digital single lens reflex (DSLR) camera. DSLRs offer larger image sensors as well as control over exposure, depth of field, colour balance, sensitivity, and graininess.

This chapter is intended to help you quickly set up the camera for a studio portrait, copy work, or product shot.

Quick Camera Setup Procedure

1. Before mounting the camera, familiarize yourself with the controls for exposure mode, ISO, aperture, shutter speed, white balance, and auto/manual focus.
2. Turn autofocus on. It’s often more accurate than the eye. But prepare to turn off for the gray card.
3. Set ISO to 200. (Higher is more light-sensitive but also more grainy.)
4. Mount the camera on the tripod or copy stand.
5. Turn on and position the studio or copy stand lights.
6. Choose an aperture for the depth of field you want—e.g., f/8 for portrait or copy work, f/16 or higher for 3D objects.
7. Compose your photo with a flat subject on the copy stand, a 3D subject on a product photography stage or illumination tent, or portrait against a backdrop.
8. Show the camera an 18% gray card, turn off autofocus, and set the exposure to get EV of 0. (Autofocus won’t work if it can’t detect edges.)
9. Set white balance on the gray card.
10. Bracket your exposures. Adjust the shutter speed ±0.3, 0.5, and/or 1 EV. Bracketing only takes a few seconds and will give you a range of exposures from which to choose on a large-screen display and with the help of levels or dot area measurements.

Camera Models

Professional and advanced amateur DSLR cameras that are useful for web photography include so-called professional point-and-shoot, mirrorless, and single-lens reflex cameras (**Figure 14-1**).

- Professional point-and-shoot cameras may be slightly larger and heavier than pocket cameras, but offer the ability to set exposure, aperture, and white balance manually. They can also capture images in camera raw format.
- Mirrorless cameras offer the same controls as DSLR cameras, but lack optical viewfinders, which saves space and weight. This type of camera is especially useful for photographers who have gotten used to composing images on-screen rather than by peering through a viewfinder.
- DSLR cameras are based on film cameras and include a traditional viewfinder. The term “single-lens reflex” comes from the camera’s use of a single lens (rather than a separate viewing lens, as in rangefinder cameras). A mirror directs light from the lens to the viewfinder and flips up, or “reflexes,” when an exposure is made, allowing light to reach the sensor.

Figure 14-1. An assortment of cameras suitable for high-quality web photography, including a canon S120 professional point-and-shoot, Pentax Q10 mirrorless, and Pentax K-r DSLR.

Camera Settings

Modes. High-end point-and-shoot, mirrorless single-lens reflex (SLR) and digital SLRs with viewfinders typically have a dial (**Figure 14-2**) with a selection of various exposure modes (**Table 14-1**).

Figure 14-2. Mode dial on a DSLR camera includes manual, aperture priority (“Av”), shutter priority (“Tv”), ISO priority (“Sv”), and program (“P”) modes, along with various scene settings. For web work manual gives the greatest control and consistency.

Manual mode is recommended for studio work. Although automated modes can be sophisticated tools, the camera could set up each shot differently. For copy, product, still-life, and portrait studio work, we want all images to look the same.

To use manual mode effectively, the exposure should be set on an 18% gray card (**Figure 14-3**) after composing the photo.

Figure 14-3. A 15% grey card (to right of table) is used to set manual exposure to 0 EV on this DSLR copy stand setup.

Table 14-1. Camera Modes

Mode	Description
M	manual mode—user sets aperture and shutter speed
A	aperture-priority mode • user sets aperture, camera sets shutter speed • good for maintaining constant depth of field
T	shutter-priority (“T” for time) • user sets shutter speed, camera sets aperture • useful for handheld work and moving subjects where you want to maintain a certain shutter speed to freeze motion or avoid motion blur that comes from handheld shots below 1/30 sec.
P	programmed mode • camera selects aperture and shutter speed
Scene	scene mode—useful for certain types of subjects, especially outdoors
Auto	Some camera manufacturers have proprietary scene-detection settings.

Autofocus. The autofocus feature of digital cameras works by detecting edges of objects. Autofocus can be very

useful, unless it detects the wrong area of the subject or if it's trying to focus on a gray card or other subject that has no apparent edges.

Autofocus can usually be set to focus on a single or multiple points. In some cases the single point can be moved around the frame, making it easier to choose the spot you want to be in-focus.

ISO. The ISO setting controls the sensitivity of the camera's light detector. Most cameras range from 100 to several thousand (e.g., 10,000 or 25600). Higher ISO = greater sensitivity = shorter exposure times = granier photos (**Figure 14-4**). Most modern cameras produce satisfactory photos at ISO settings of 200–1,000.

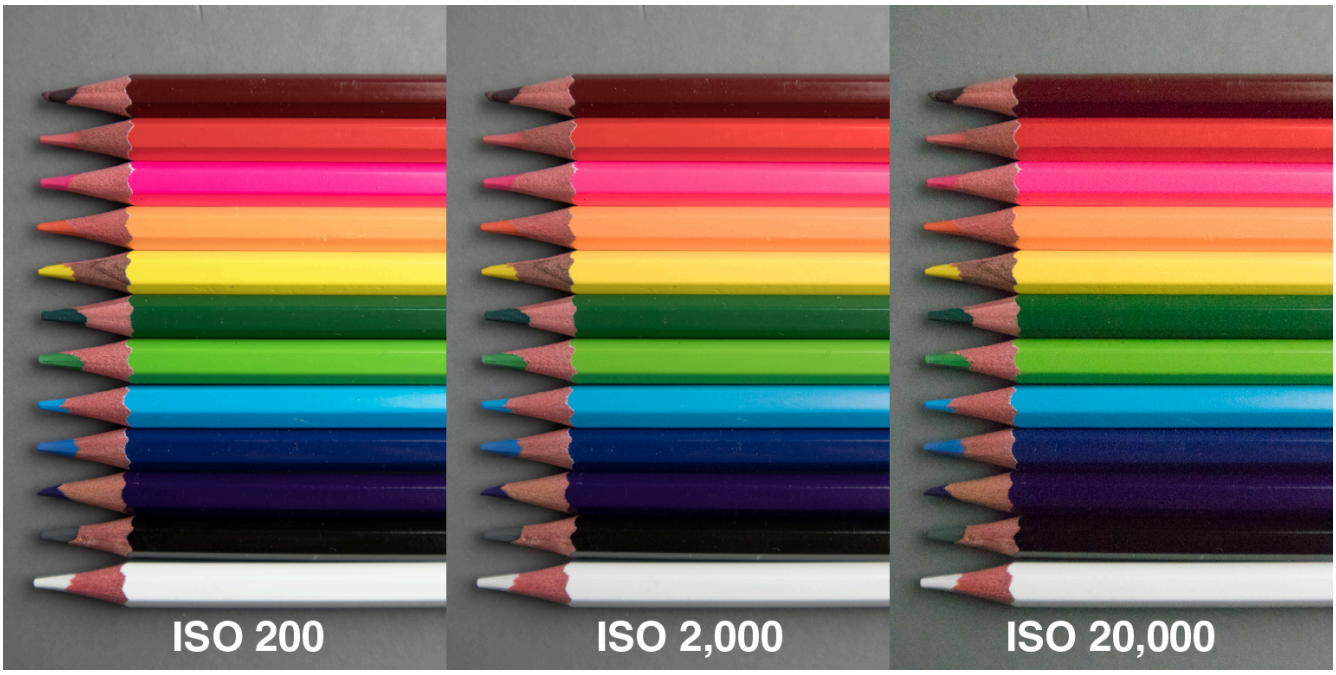


Figure 14-4. Three identical photos taken with a Canon EOS M100 mirrorless camera at ISO 200, 2000, and 20000. When examined in detail the ISO 200 photo can be seen to be the smoothest while the ISO 20000 photo is the grainiest.

Higher ISO settings are useful for outdoor photos when there is low light. The challenge of low light is maintaining a shutter speed above 1/30 sec. to avoid handheld motion blur, while also setting a small enough aperture to get good depth of field.

Tripod. A tripod helps keep the camera steady and avoid motion blur from camera shake. Quick-release tripods (**Figure 14-5**) have a detachable plate that screws into the tripod socket on the bottom of the camera and then clamps to the tripod.

Figure 14-5. Quick-release tripod mount includes a plate that screws into the camera mount and then clips onto the tripod, allowing for rapid on-and-off operation.

Copy stands also have a screw mount that holds the camera steady. However, for some shots, touching the camera to make the exposure could cause it to move. For sensitive shots the camera can be triggered using its built-in timer or with a remote control.

Zoom. Unless your’re using a specialized or macro lens, your camera probably has a zoom lens that lets you change the focal length to crop the scene.

For copy and product photography, position the camera or set the zoom lens so the subject takes up most of the frame, without cutting off any of its parts (**Figure 14-6**). This ensures that the largest number of megapixels is used for the subject.

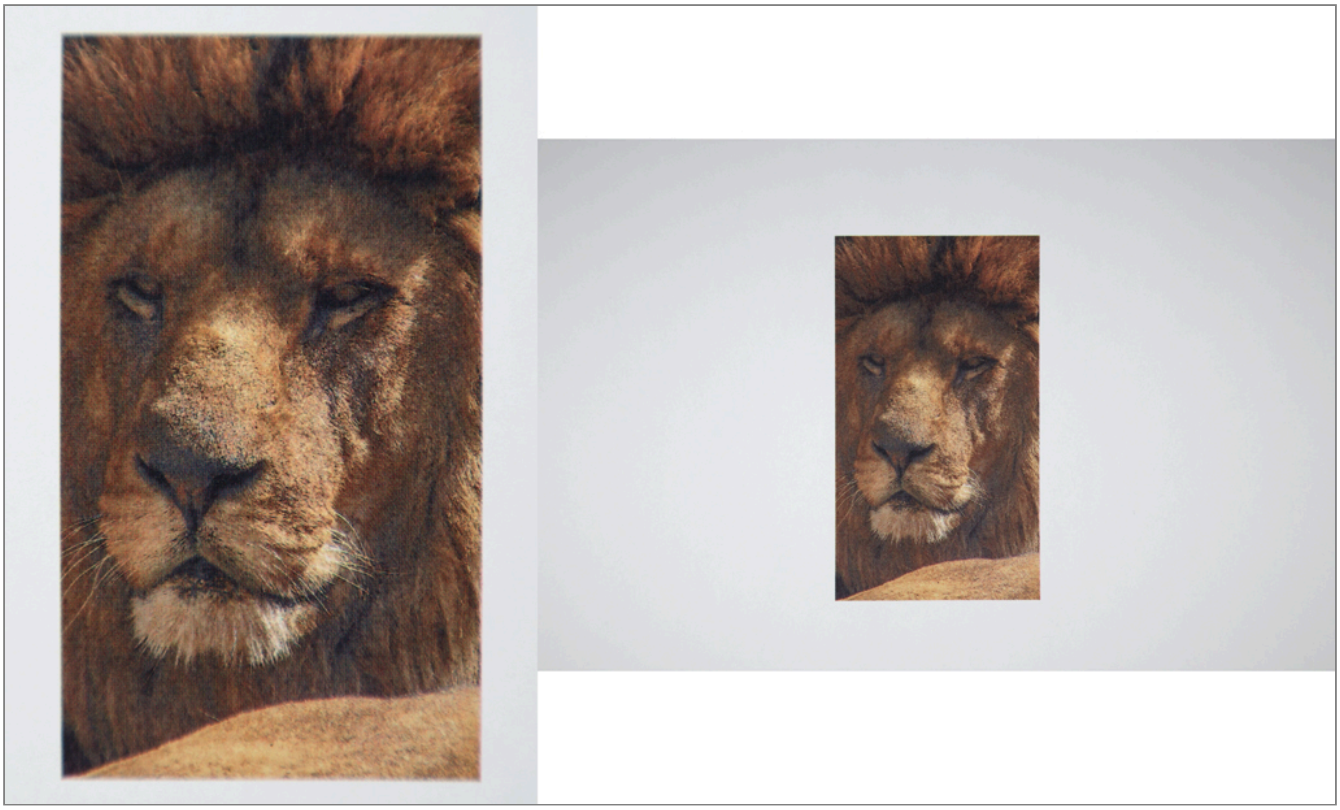


Figure 14-6. Good composition (left) takes up most of the camera's sensor area with the subject. (Image courtesy of Pixabay.)

Aperture. The aperture is an expandable iris diaphragm that controls how much light is admitted to the camera (**Figure 14-7**). The aperture controls depth of field and the focus from front to back (**Figure 14-8**). The scale in practice is 1.4–32, where higher number is smaller aperture and greater depth of field.

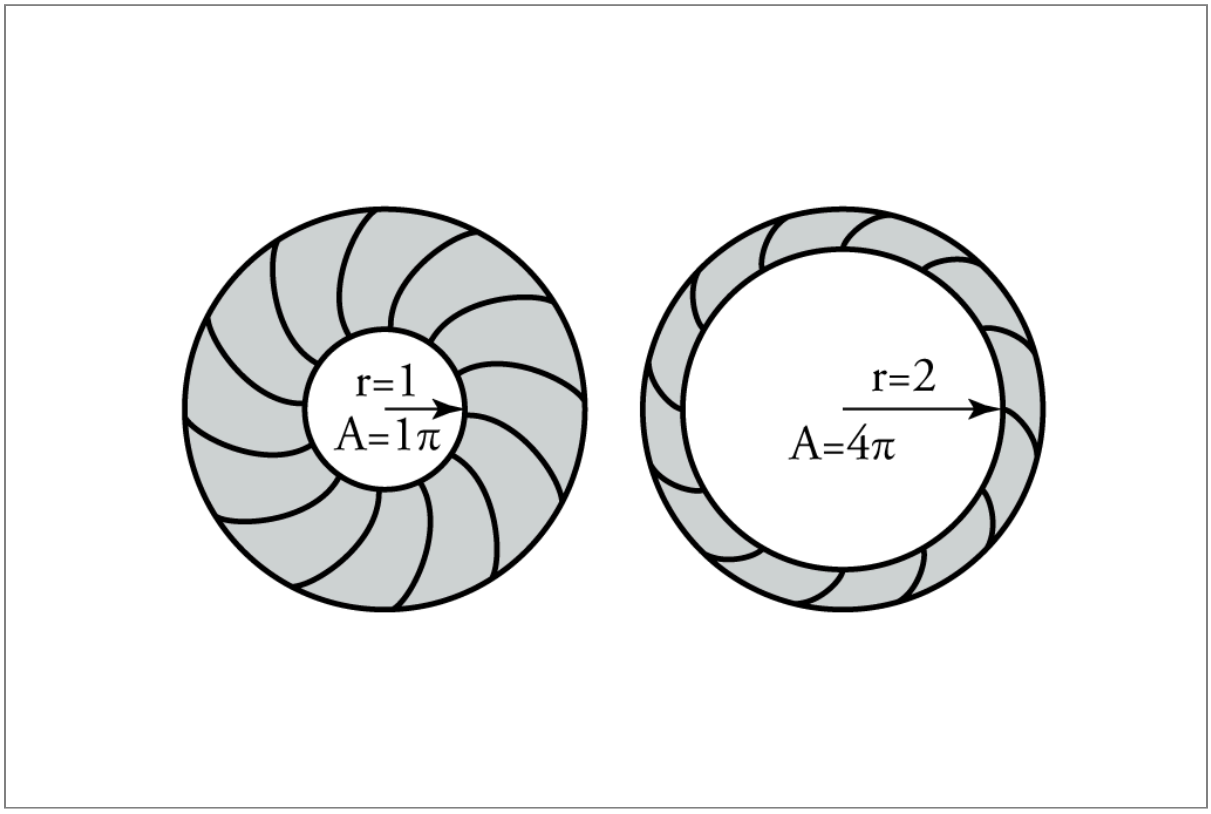


Figure 14-7. The camera's aperture controls the amount of light admitted to the sensor and also affects the depth of field, the depth of scene that's in focus from front to back. Smaller apertures (left) admit less light but provide greater depth of field.

Figure 14-8. Miniature orchid photographed with an wide aperture (f/8, left) and narrow aperture (f/32, right) providing greater depth of field.

Shutter speed. The length of time light is exposed to the sensor. Controlled by a curtain in front of the sensor. Typically ranges from 1 sec. to 1/4000 sec.

When handheld, speeds slower than 1/30 sec. make produce camera shake that blurs images.

Exposure bracketing. In addition to the technically correct exposure of EV 0 on an 18% gray card, bracketing refers to exposing additional photos, ±1/3, 1/2, and/or 1 EV (**Figure 14-9**).

To bracket the exposure, increase and decrease the shutter speed while leaving the aperture at the same setting so depth of field is not affected.

Looking at a small, non colour-managed photo on the camera's display is not as accurate as looking at the same image on a large, colour-managed computer screen.

And, on the computer screen, the highlight and shadow values can be measured with the Info tool as levels or percent dot area.

Figure 14-9. Bracketed series of exposures include -0.75 EV (left, underexposed), 0 (center, correct exposure), and +0.75 EV (right, overexposed). When viewed on a small non-colour-managed screen, the differences may be hard to see.

Saving files. Modern professional point-and-shoot, mirrorless, and DSLR cameras can typically be set to save photos in Camera Raw and JPEG Fine or JPEG Medium format.

Camera Raw. This format records the maximum number of megapixels that the camera is capable of. Camera Raw be saved in proprietary or .DNG (digital negative) format. Examples of proprietary formats include Nikon NEF, Canon CRW, and Sony ACR.

Camera Raw files are best opened in Photoshop but can be batch-covered using Adobe's free DNG Converter program. A disadvantage of Camera Raw is the large file size. The advantage of this is that when opened in Photoshop, multiple photos (e.g., a bracketed series) can be selected and have the same crop or other settings applied before opening. You can also choose the size of the photo you want, in megapixels or pixel dimensions.

JPEG. This format is compressed, meaning it reduces the file size to save space and speed up copy and transmission. Higher-quality JPEG settings (e.g., 9–12 on a scale of 0–12) may be indistinguishable from uncompressed TIFF format.

Fine and Normal indicate quality level and corresponding file size of the JPEG. Normal is useful if you want to fit lots of photos on a small memory card. However today most memory cards are 8, 16, 32, or 64 GB and can hold hundreds of Camera Raw and JPEG-Fine photos.

Mounting and Lighting Equipment

Several types of camera accessories are useful for studio shots used in graphic arts, including 2D copy, 3D product, still-life, and portrait shots.

Copy stand. A copy stand (**Figure 14-10**) consists of a horizontal camera mount and two lights held at a 45° angle to the subject, to minimize light reflection into the lens. It is useful for 2D shots such as art work, drawings, printed photos, and 3D subjects where only one side needs to be captured.

To use a copy stand, first familiarize yourself with the camera's settings, including mode, shutter speed, aperture, ISO, white balance, and auto/manual focus, as they may be difficult to find after mounting on the stand.

Check that the camera is level after mounting. A level, such as a bubble level or the iOS Compass app (with level as the second screen) may be helpful.

Figure 14-10. A copy stand is useful for photographing flat objects. Two lights at 45° angles help reduce glare from the subject's surface.

Tripod. A tripod is useful for 3D products, still lifes, and portraits and helps avoid motion blur. Quick-release tripods (see **Figure 14-5**) have a detachable plate that screws into the tripod socket on the bottom of the camera and then clamps to the tripod.

Studio Lights and Diffusers. Studio lights are necessary to get the proper exposure indoors. Lighting types include quartz-halogen "hot lamps," xenon strobes, fluorescent, and LED.

Other ways to categorize lighting including “constant” vs. “momentary illumination” and “hot” or “cool”. Constant illumination (quartz-halogen, fluorescent, and LED) makes it easier to view and compose the scene, while strobes provide the most powerful light. Cool lights (fluorescent and LED) are safest to handle.

Light diffusers (**Figure 14-11**) are necessary to break up harsh light. Diffusers can include translucent cloth or plastic, or reflective umbrellas.

Figure 14-11. Studio lights can be equipped with various types of diffusers, including transmissive “soft boxes,” as in the photo, and reflective umbrellas.

Product Photography Tents. These are useful for illuminating small products or still-lifes of related items. They consist of a background with surrounding lights (**Figure 14-12**).

Figure 14-12. A light tent provides diffuse light for product photography.

Camera Accessories

UV Haze Filter. A UV haze filter is a clear filter that cuts out UV light which helps protect camera lens. If it gets scratched, the filter is a lot cheaper to replace than the lens.

Polarizing Filter. This filter acts like polarized sunglasses for the camera. It filters out reflected light from non-metallic surfaces, reducing glare and increasing colour saturation (**Figure 14-13**). Polarizing filters are generally dark and reduce exposure by 1 EV. They may also affect white balance. For best results, calculate exposure and reset white balance after attaching the filter.

Figure 14-13. Photo of an orchid taken without (left) and with a polarizing filter.

Lens hood. A lens hoodattaches to the front of the lens (**Figure 14-14**). It helps reduce glare that could come from stray light entering the lens, which could reduce the contrast of the photo.

Figure 14-14. A lens hood reduces reflected glare onto the lens, which increases the contrast of photos and avoids artifacts from stray light.

Macro lens. Macro lenses (**Figure 14-15**) are especially made for photographing items up close. The magnification of the macro lens is expressed as a ratio of the actual scene or subject size to the size of the reproduction. A 1:1 macro lens can photograph items actual size.

Figure 14-15. A macro lens (left) is useful for photographing items up close, typically with a 1:1 reproduction ratio, so the item is captured in its actual size (right) on the sensor.

File Types

Most professional point-and-shoot, mirrorless, and DSLR cameras can record images in Camera Raw and JPEG format.

Camera Raw is an uncompressed format that saves the maximum number of megapixels (MP) of which the camera is capable. That is, if a camera is capable of 12, 16, or 24 MP, Camera Raw images will be recorded at the maximum resolution.

Camera manufacturers have proprietary Camera Raw formats for each camera model. To open these images in Photoshop or Preview, Adobe and Apple must update their converters for each new camera that comes on the market.

Upon opening in Photoshop, the files can be saved in other formats useful for print or web publication. For print applications, the most useful formats are PSD, TIFF, and JPEG High Quality. For Web applications where download speed is paramount, JPEG, PNG, and GIF are useful.

Table 14-2. File types

Type	Description
Proprietary Camera Raw	Unique to each digital camera, these can be opened in Preview or Photoshop, provided that Apple and Adobe have updated to the necessary file converter.
DNG Camera Raw	Digital Negative, a standard, universal Camera Raw format. Can be opened in Photoshop and other applications.
JPEG	A “lossy-” compressed file format where some data is discarded to reduce file size. Highest quality format can be difficult to distinguish from uncompressed TIFF.
PSD	Photoshop Document, Adobe’s proprietary version of TIFF. Can be read by InDesign and Illustrator but not web browsers.
TIFF	Tagged Image File Format, a standard for uncompressed files. Can be read by most word processing and page layout programs and also by web browsers if in RGB mode.
PNG-24	Portable Network Graphic, 24-bit colour. A continuous-tone format of PNG for photos. Supports transparent backgrounds for Web publication.
PNG-8	PNG, 8-bit colour. An indexed colour mode suitable for flat art or “comic-book” colour. Also supports transparency.
GIF	Graphic Interchange Format. An indexed colour format for flat or “comic-book” colour, mainly for publication on the Web.

Adobe DNG Converter

This free program from Adobe does batch conversion of proprietary Camera Raw images to standard Digital Negative (DNG) format (**Figure 14-16**).

To use DNG Converter:

1. Copy proprietary Camera Raw images from your camera’s SD card to a folder on your computer.
2. Start DNG Converter and browse to the input folder.
3. Designate a folder to contain the converted DNG files.

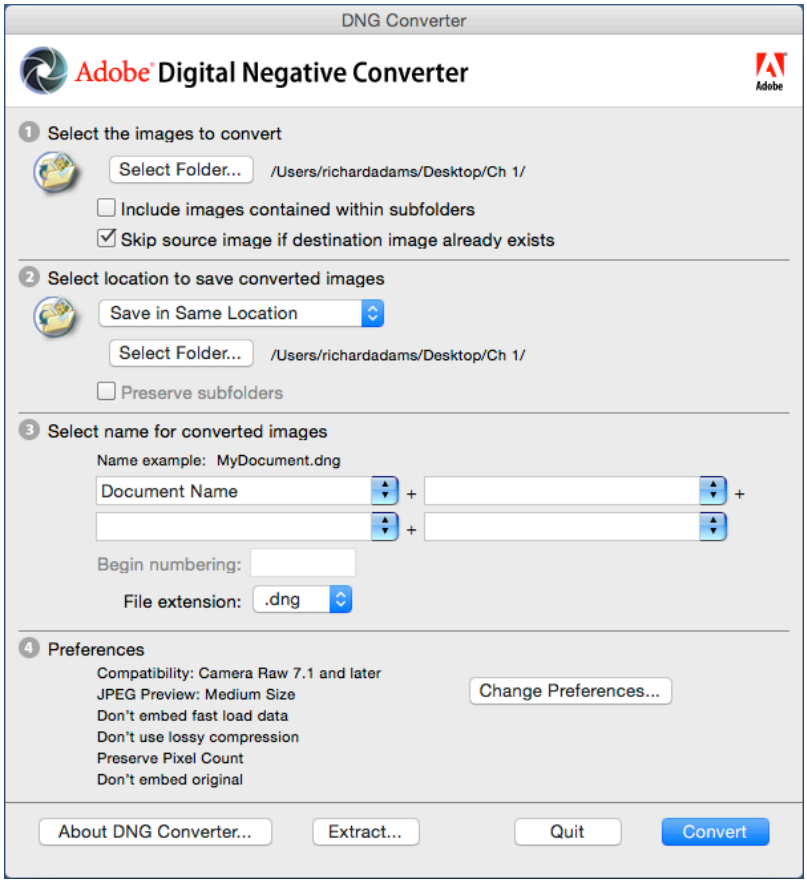


Figure 14-16. Adobe's free DNG Converter program is useful for converting camera raw (including proprietary camera files) into standard digital negative (DNG) and other formats.

Chapter 15 - Photoshop for Web and Cross-Media

Introduced in 1989, Photoshop is a market-leading image editing program that is useful in web design for opening, formatting, editing, and saving images for the web.

Image Capture

The best way to capture high-quality images for the web is with a digital single-lens reflex (DSLR) camera, whether mirrored or mirrorless. Today's semi-professional and advanced amateur cameras typically capture 20–36 megapixels (MP), and the megapixel capacity increases as new cameras are introduced every year.

Camera raw format captures the full megapixel and density range of the camera, however, such large images are seldom necessary for the web. In fact, they can be detrimental to loading times. Camera Raw includes both proprietary formats for each camera manufacturer and model, as well as the universal digital negative (DNG) format.

When opening camera raw format in Photoshop's Camera Raw plugin (**Figure 15-1**), web designers should think about the maximum image size they will need, then use the image resizing feature to get the desired size. The site websitedimensions.com says that 35% of web surfers use a browser window that's 1366×768 px, while 20% use 1920×1080. If this is true then the maximum size window would be 2 MP (1920×1080) and a full-size image would not need to be larger than that. It may be useful for the designer to think about the anticipated maximum size of an image as being a full-page, half-page, or quarter-page image. A quarter-page image would need to be 960×540 or 0.5 MP.

1. Set camera exposure, white balance, and depth of field for optimum photo.
2. Record in camera raw format, bracketing exposures.
3. Open photos in Camera Raw plugin, select correct exposure, set image size, and open.
4. Edit as required.
5. Export to JPG or PNG using Export > Save for Web.

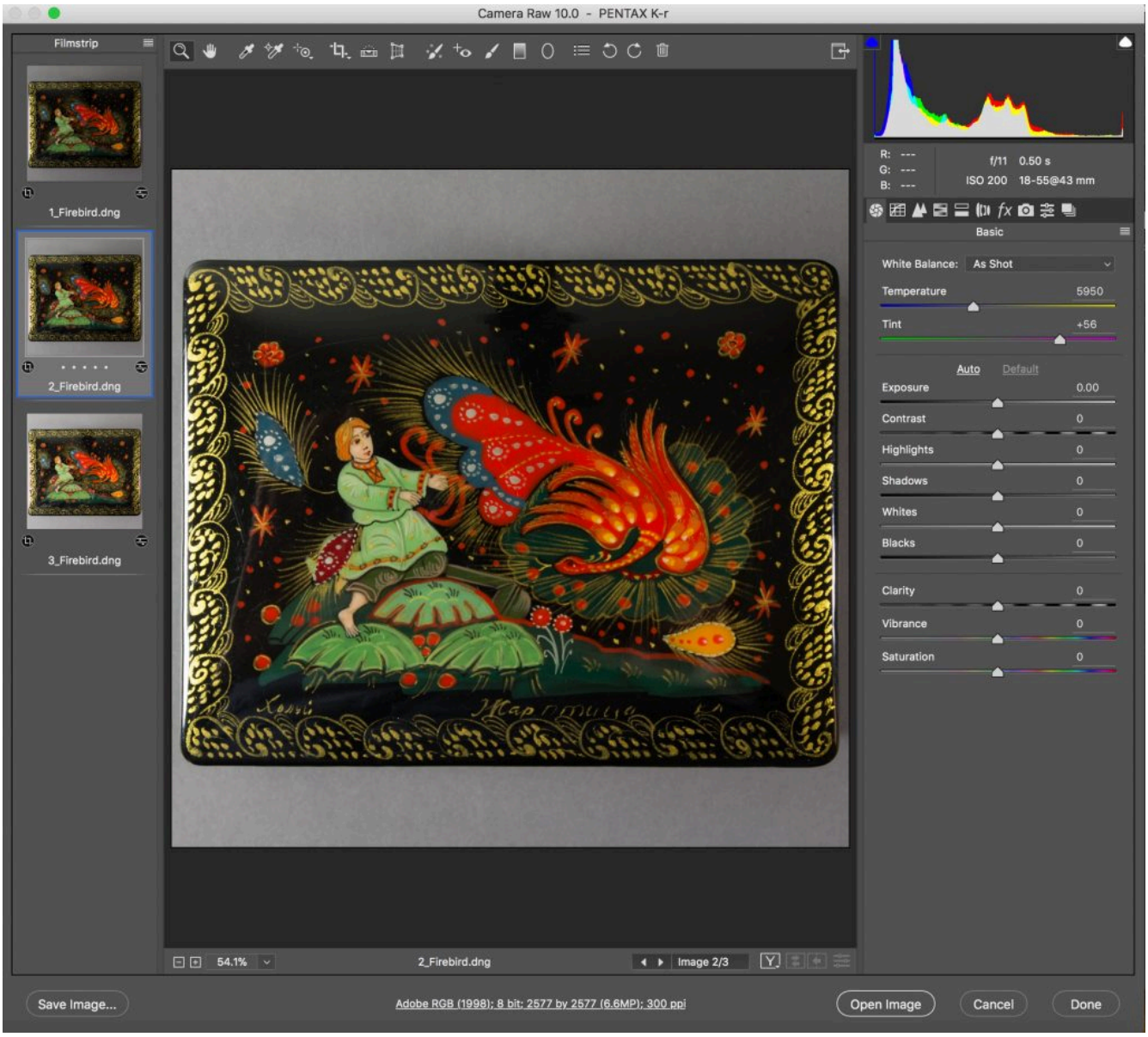


Figure 15-1. Photoshop's Camera Raw dialog box offers the opportunity to open several photos camera raw files at once, make settings, and save in standard formats. In this case a bracketed series of exposures (i.e., one slightly too dark and one too light) was taken with a 24-MP camera. The best photo was opened at a resolution of 72 ppi and size appropriate for the web. (Adobe® PhotoShop ® screenshot(s) reprinted with permission from Adobe Systems Incorporated. Adobe® Photoshop® are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.)

"Toning" Images

If you have a bracketed series of exposures, several features of Photoshop are useful for choosing the best image and preparing it for publication.

Obviously it's easier to see the quality of a photo on a large-screen computer monitor than on a camera's 3-inch

display. However, Photoshop and other programs also allow you to compare photos “by the numbers” by reading highlight and shadow values.

The lithographic printing industry took a systematic approach to image toning, based on the largest and smallest halftone dots that a press can typically reduce. Measured on a scale of 0–100% dot area, most presses can reproduce 3% highlight dot and a 97% shadow dot.

For images on the Web, monitors display colour on a scale of 0–255 levels, where level 0 is black and 255 is white. Applying the 3-97% minimum-maximum to the levels scale: 3% of 255 is 7.65, or 8 when rounded; while 97% of 255 is 247.35, or 247 (**Table 15-1**).

Table 15-1. Highlight and Shadow Dot Values

	Highlight	Shadow
% Dot Area for Print	3%	97%
Levels for Web	247	8

To select the best exposure from a bracketed series, make sure the file has no levels above 247 or below 8. After selecting the best image from a bracketed series, you can further “tone” the image (**Figure 15-2**) by adjusting highlight, shadow, midtone, gray balance, and colour.

Figure 15-2. Original (left) and toned (right) images. On the right-hand image, highlight and shadow were set at 247 and 8 levels, respectively, to bring out the maximum contrast in the image.

Making Transparent PNG Files

If your image contains one or a group of subjects that you want to publish on the Web without the background, save the image as a PNG file with transparent background (**Figure 15-3**).

First, make a selection path around the object(s) you want to isolate from the background. Then right-click on the selected subjects and choose “New Layer from Cut.” Select the background layer and set its transparency to 0%. Save for Web in Photoshop, check the Transparency option, and verify in the preview.

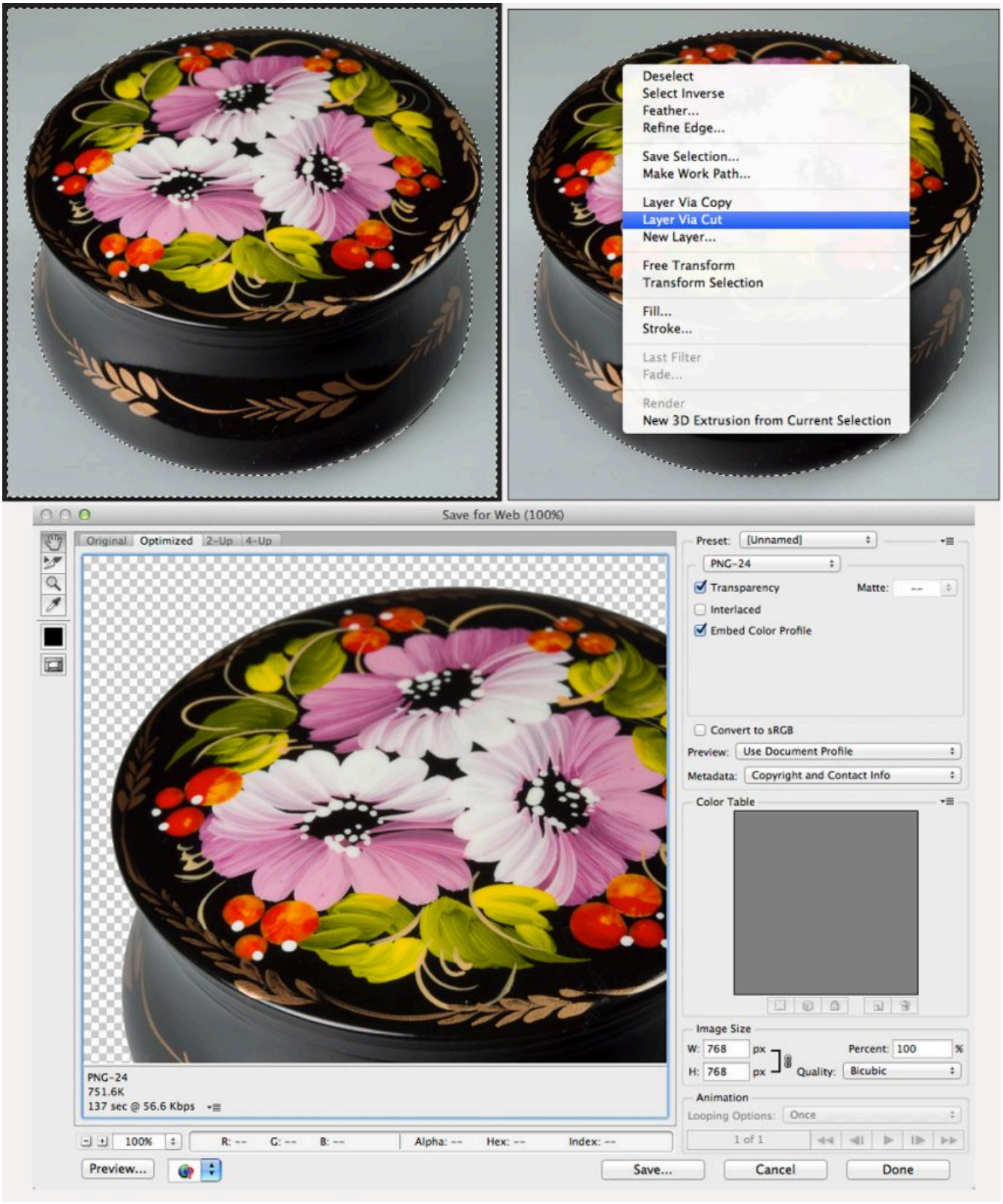


Figure 15-3. Making a transparent PNG file by selecting the object, bringing it to a new layer using “New Layer via Cut,” deleting the background, and Export > Save for Web with Transparency checked.

Saving Images for Web

Once an image has been opened and edited in Photoshop, it can then be saved in a format suitable for web (JPG for photos and PNG when transparency is required) using the Export > Save for Web feature. This dialog box (**Figure 15-4**) includes settings for image format, transparency, size, and resolution.

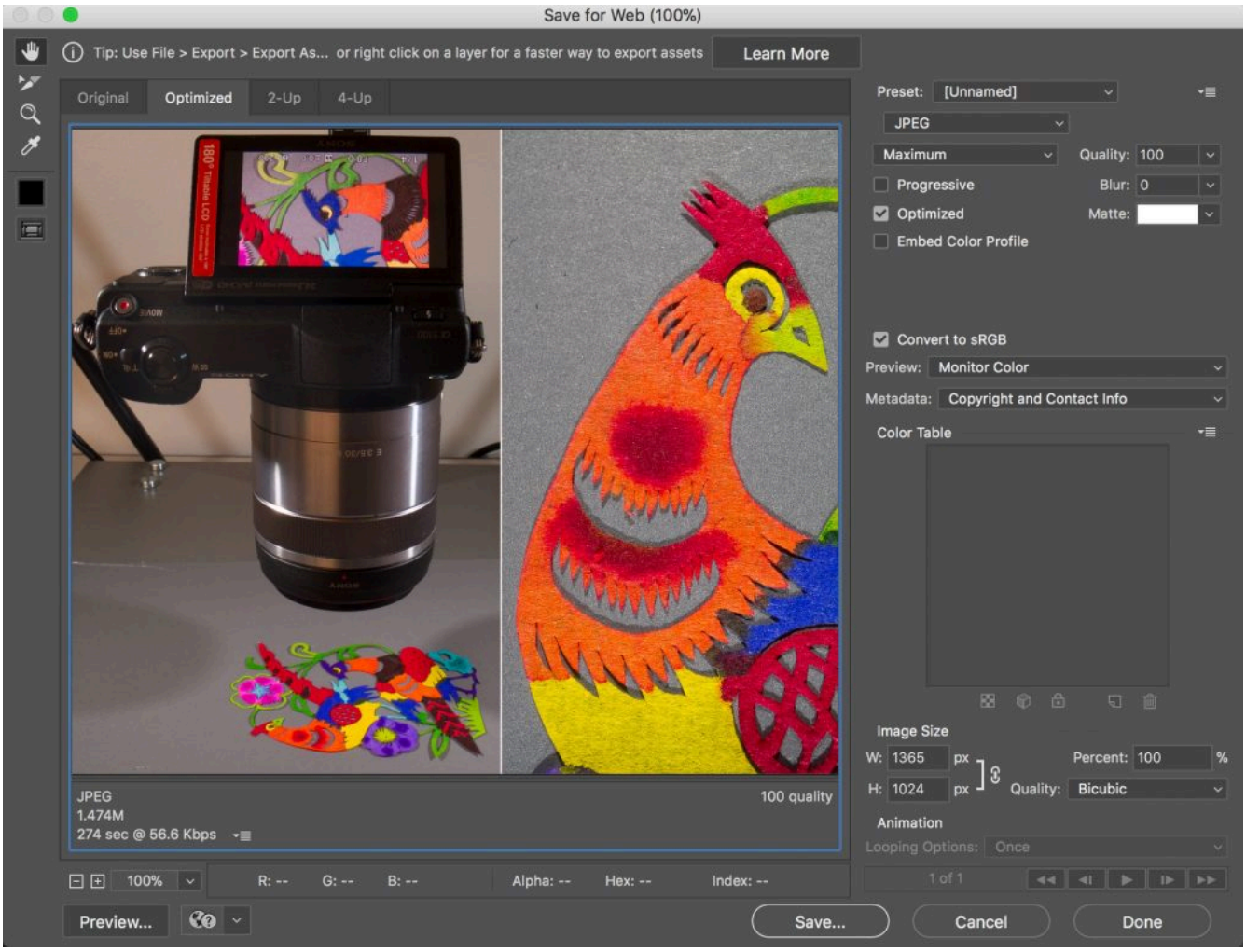


Figure 15-4. Photoshop's Export > Save for Web dialog box offers the opportunity to customize the image format (upper right corner) and size (lower right) and also includes a prediction of download time (lower left) at various internet bandwidths.

Non-Destructive Editing

As a bitmap image-editing program, Photoshop enables users to change the tone and colour, along with the pixel content, of images. This is known as *destructive editing* because the original characteristics are altered.

Non-destructive editing is done through layers and masks and preserves the original pixel information.

According to Jason Lisi, Associate Professor in Ryerson's School of Graphic Communications Management, who teaches a faculty-wide open elective course in Photoshop, there are three reasons why non-destructive editing is useful:

1. *Preserving the original.* Since non-destructive editing preserves the original pixels, you can always go back to the original image if you don't like the results of your edit.
2. *Making a record.* Non-destructive edits provide a step-by-step record of changes, so that you can later go back to them. This could be useful when editing for a client, to show what edits were made and justify the time spent making them.
3. *Working faster.* Editing via layers and masks allows you to organize your edits, see what you are doing, and work faster and more efficiently.

Tools for Non-Destructive Editing

Layers. A fundamental concept of non-destructive editing is to apply edits to Photoshop Layers. Adding a layer is like placing a clear plastic overlay on top of a photographic print, and making the edits on the plastic. The effects of the edit layer can be applied to the layer(s) below, then turned on and off to visualize the photo with and without the edits.

Layer Masks. A Layer Mask accompanies a Layer and can hide or show part or all of the effects of the Layer.

Adjustment Layers. This type of Layer applies Image > Adjustment effects, such as Brightness/Contrast, Levels, Curves, and others, via a Layer, so that changes are non-destructive.

Layer Styles. A Layer Style adds effects to another layer, such as a drop shadow, emboss effect, gradient, or other effect.

Smart Objects. These include placed images and image components whose original pixel data is preserved after edits. Let's say you place a logo or other image into an existing image and then want to reduce its size. If first converted to a Smart Object, Photoshop will retain all the pixels in the original logo so that it could be enlarged later.

Smart Filters. These are Filters applied to Smart Objects so the effects of the filter can later be removed or modified.

Content Awareness. A form of artificial intelligence, content awareness examines neighbouring pixels and uses them to fill in spaces, move objects, and resize images (**Figure 15-5**).

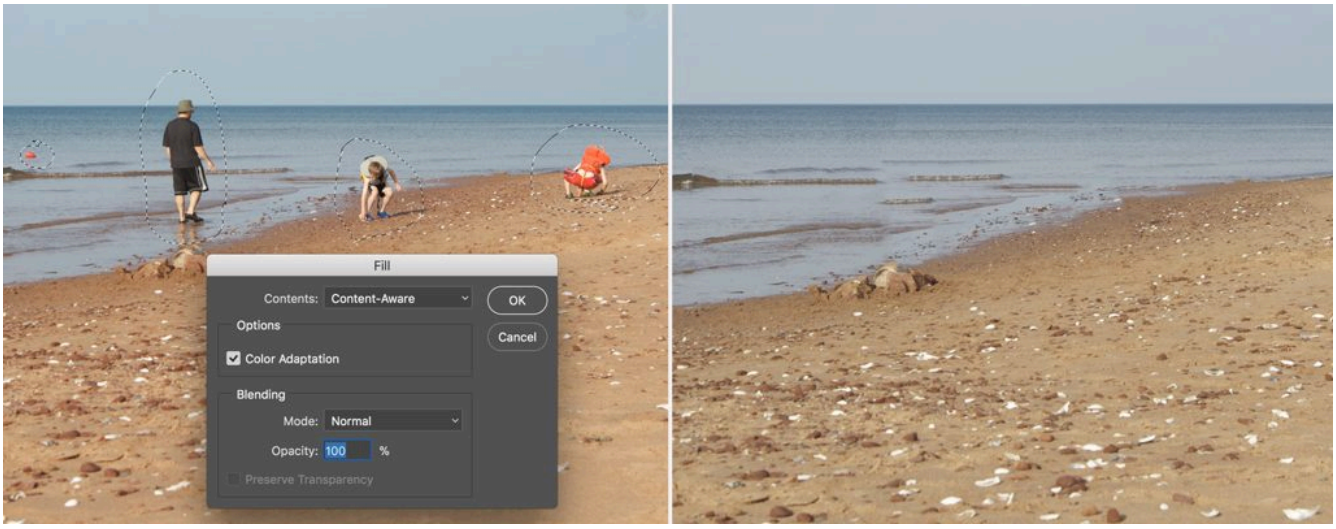


Figure 15-5. People were removed from this beach scene using Photoshop's Content Aware Fill feature. Marquis were drawn around each object to be removed, then Edit > Fill was selected and set to Content Aware. (Photo courtesy of Jason Lisi)

Tips for Non-Destructive Editing

10. Duplicate a Layer. Let's say you open a digital photo in Photoshop and want to make some edits. The fastest way to edit non-destructively is to duplicate the Background Layer into which the photo is opened. Simply drag the layer to the New Layer icon in the Layers palette to duplicate it. Or select the layer and choose Duplicate Layer from the Layers menu or the upper-right popup menu in the Layers palette.
11. Retouch through a Layer. When retouching defects, artifacts, or undesirable features of an image, the Clone Stamp, Healing Brush, and Spot Healing Brush tools can be applied through a Layer (**Table 15-2**). Simply check "Sample All Layers" or choose "Current and Below" from the popup in the Properties menu. (Not all tools work through layers, see **Table 15-3**. In this case, duplicate the layer before retouching it.)

ToolPurpose

Table 15-2. Retouching Tools that Work through Layers

Clone Stamp Tool	Duplicate pixels from another area
Healing Brush Tool	Automatically fix artifacts in image by painting over them
Spot Healing Brush Tool	Automatically fix artifacts in a specific area

ToolPurpose

Table 15-3. Retouching Tools that Don’t Work through Layers (duplicate layer first)

Content-Aware Fill	Fills an area using pixels within proximity
Content-Aware Move	Moves selected pixels and replaces them automatically
Content-Aware Scale	Scales selected pixels while automatically filling empty pixels

10. Use an Adjustment Layer to Change Tone or Colour. To improve contrast, brightness, highlight, shadow, gray balance, selective colour correction, and other adjustments, apply an Adjustment Layer to make these changes non-destructive.
11. Apply a Layer Effect. Layer Effects make it easy to add a colour blend, metallic effect, drop shadow, and many others in a non-destructive manner.
12. Use a Layer Mask. A Layer Mask hides, shows, or alters the appearance of its associated layer. Example: If you apply a colour fill to a layer, the colour can be selectively applied using a Layer Mask. Painting or filling with black negates the effect of the associated layer, while painting with white or the Eraser Tool reestablishes the effect.
13. Use a Smart Layer for Filters. Photoshop features dozens of filters that can add noise, sharpen, smooth, and otherwise alter the appearance of images. Applying them as Smart Filters to a Smart Object preserves the original appearance of the object or layer.

Table 15-4. Layer Types in Photoshop

Type	Purpose
Layer	Holds visual contents
Layer Mask	Obfuscates content from layer
Layer Effect	Applies an effect to the later
Adjustment Layer	Changes the visual properties of the layer
Overlay	Blends layer content

10. Place Smart Objects for Resizing. When another file, such as a logo, is placed into an existing file as a Smart Object and resized, Photoshop retains all the pixels in the original. This enables the size to be readjusted without loss of information.
11. Use Content-Aware Scale to Resize. One of several content-aware features, Content-Aware Scale uses artificial intelligence to resize images while retaining the original scene objects.
12. Use Content-Aware Fill to Remove. This feature is useful for removing objects from scenes, such as people on a beach. Select the object(s) with one of the selection tools, then apply Edit > Fill > Content-Aware.
13. Use Content-Aware Move to Move. If you want to relocate an object instead of deleting it, the Content-Aware Move tool examines surrounding pixels near the source and destination to make the

moved object(s) blend in with the new background.

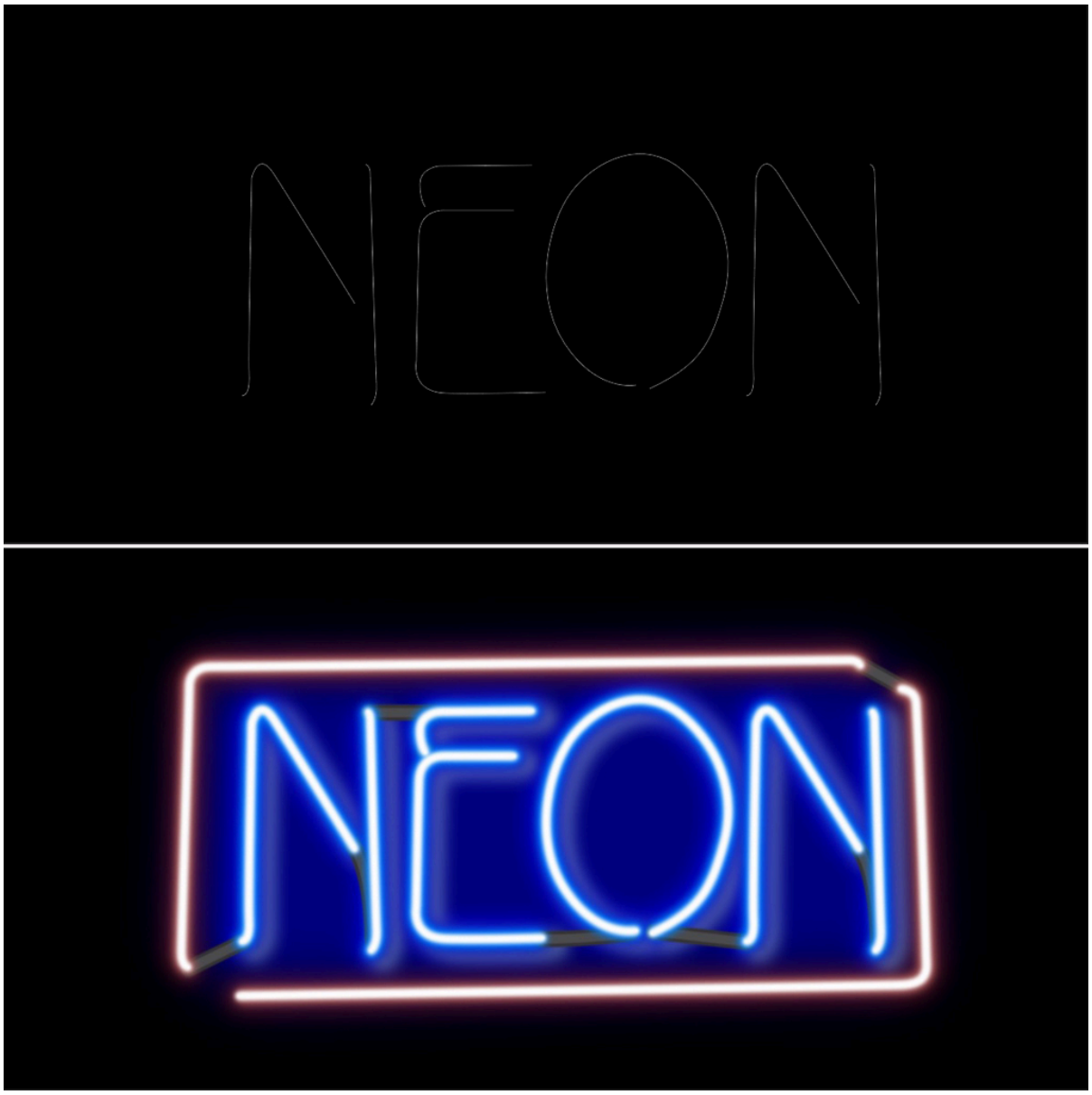


Figure 15-6. This neon type effect was created from a path using Layer Styles.

Chapter 16 - Apps for iOS

Apps for the iPhone and iPad are written with Apple's free Xcode program for MacOS, which can be downloaded from the App Store. Programs need to be written on a Macintosh computer. At present they cannot be written on an iPhone, iPad, or Windows computer.

In 2017 Apple introduced its "Everyone Can Code" initiative, which includes educational material for colleges and universities, and a free iBook, *Everyone Can Code: App Development with Swift*.

Numerous app developers have also published videos of varying lengths on YouTube and Lynda.

Orientation to Xcode

The Xcode interface is complex and detailed. As many as eight windows and palettes can be displayed simultaneously, making a large-screen display almost essential for working efficiently with the program.

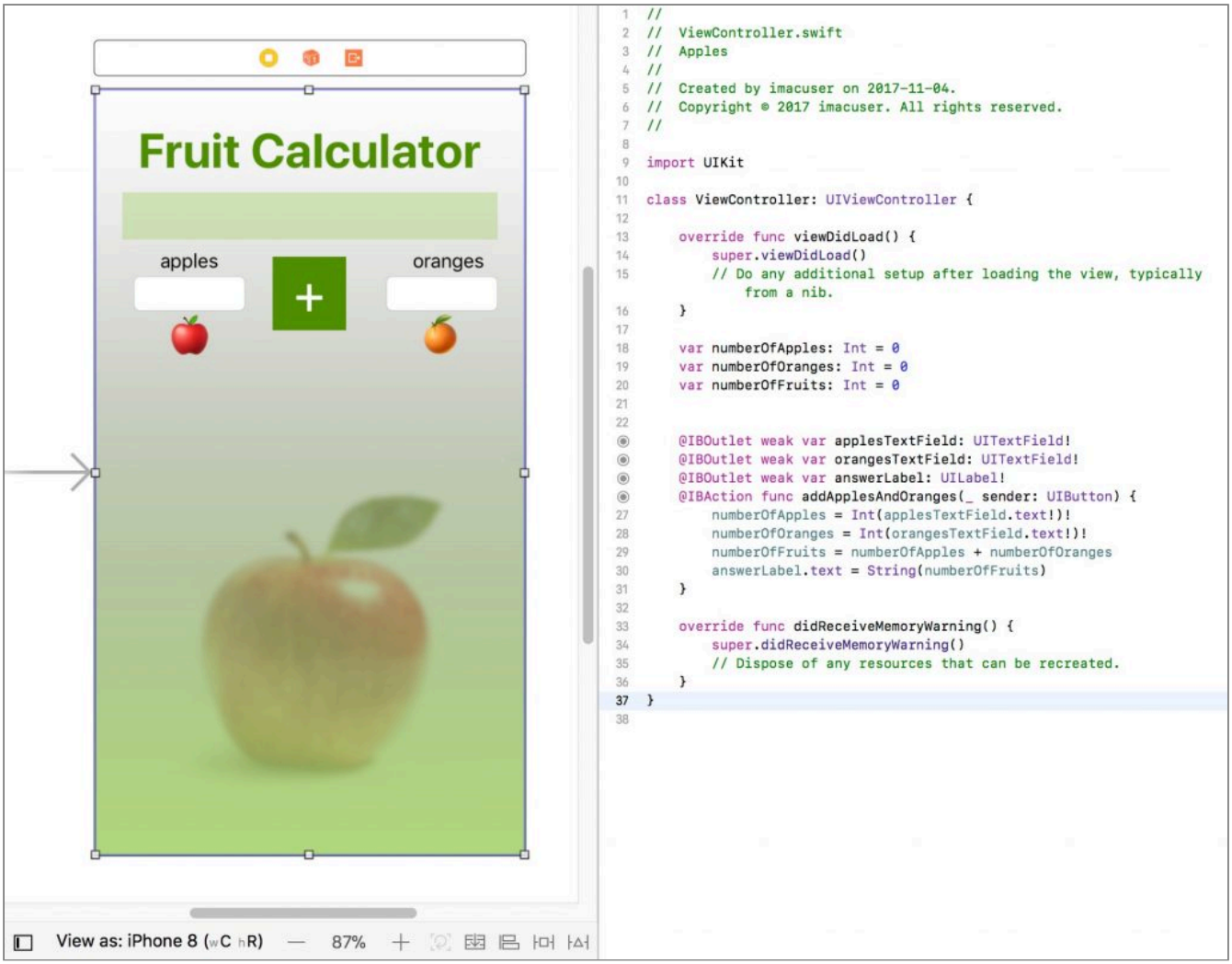


Figure 16-1a. *Interface of Apple Xcode, the MacOS program for writing apps for the iPhone and iPad. The Storyboard with screen layout is shown at left and the ViewController with code at right.*

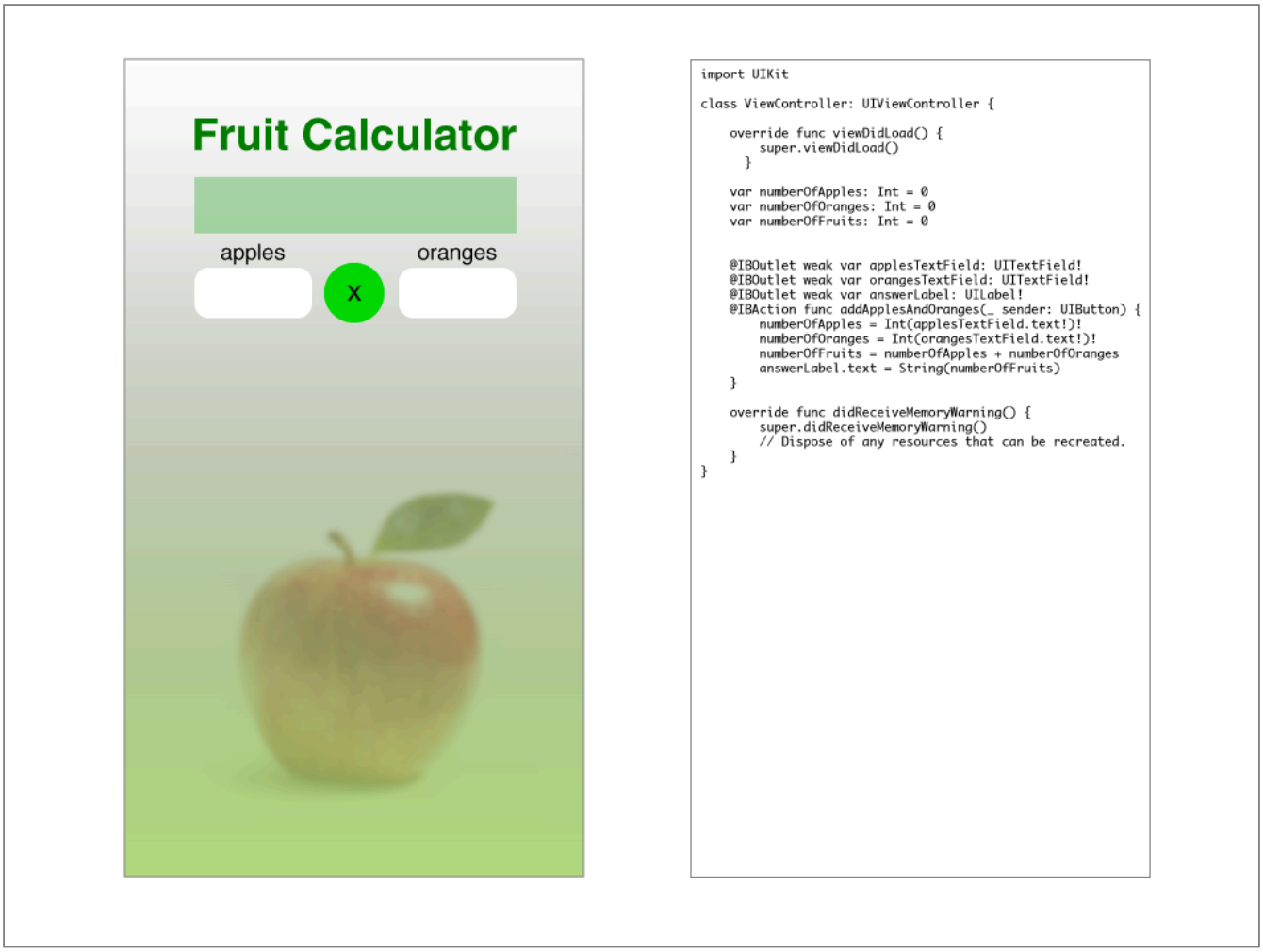


Figure 16-1b. Mockup of an app in Apple Xcode, the MacOS program for writing apps for the iPhone and iPad. The Storyboard with screen layout is shown at left and the ViewController with code at right.

An app consists of several files that contain the graphical interface and programming statements. The Storyboard (**Figure 16-1**) provides a simulation of an iOS screen, similar to the page on a page layout program. The Storyboard can contain various objects that allow for user interaction (**Table 16-1**). These are connected to programming statements in the ViewController, a separate file with lines of code that can be displayed next to the Storyboard. Elements in the Storyboard are connected to the ViewController by right-clicking and dragging a connection line. The connected elements are designated with a dot in a circle.

Xcode contains a built-in iOS simulator, called the Simulator (**Figure 16-2**), that can be set to look like any recent model of iPhone or iPad.

Table 16-1. Xcode Objects

Object	Description
text field	allows readers to enter text or numbers from the keyboard
label	used for labeling parts of the screen, such as text fields and buttons, and for displaying information output by the program
button	used to initiate actions that are programmed into ViewController functions
variables	used in the ViewController to store data and numbers; include four types—string, integer, floating point, and double precision.

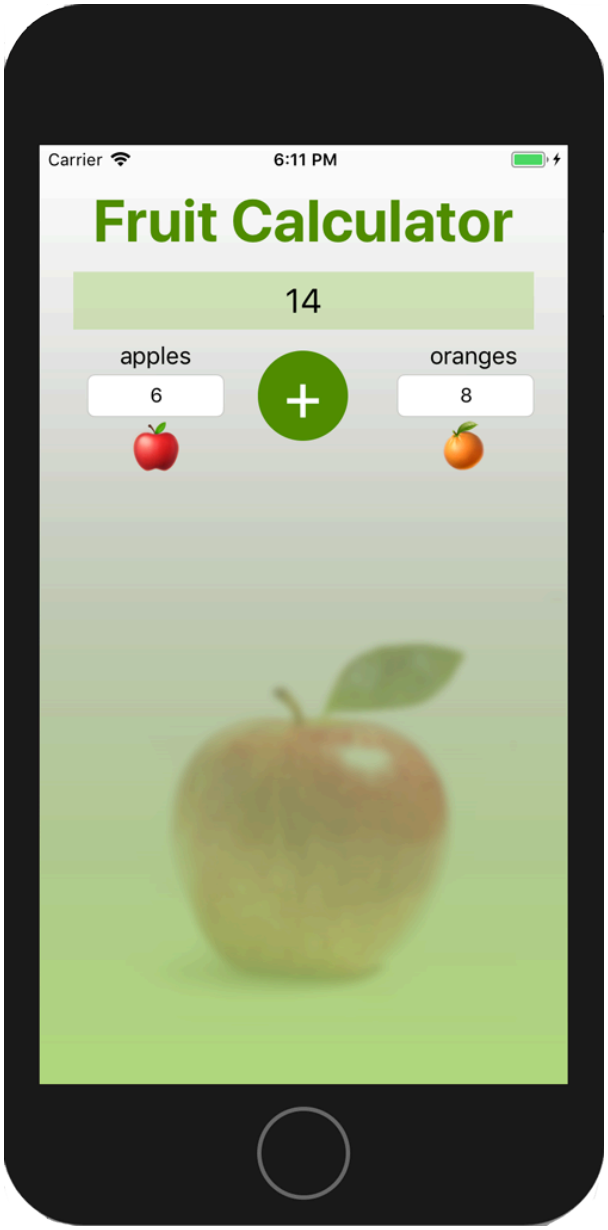


Figure 16-2. *Xcode's built-in Simulator can emulate any recent iPhone or iPad model on the Macintosh screen.*

Program Example

To provide an example of a program and the procedures used to create it, we will list the steps necessary to creating the "Fruit Salad" app shown in **Figure 16-1**, developed by Dr. Abhay Sharma for his Advanced Technology class at Ryerson University. This program accepts numerical input from readers, who type in the number of apples and oranges respectively into text fields. When readers touch the addition button, a function reads the numbers into variables, calculates the total number of fruits, and displays the total via a label.

Design the Interface

1. First we recommend designing the app's interface by sketching it on paper or using a free mockup site such as Balsamiq.
2. After designing the interface, make a list of all the text fields, labels, and buttons that you will need. Each will require a unique name in the program. Thinking of names of more than a few object variables can be confusing, especially when they need to be paired with data variables.
3. In the list of objects, include the variables that you plan to use to store numbers and text for calculation or output.

Place Objects on the Storyboard

1. Place objects on the Storyboard, including labels for the app title, field labels, and output fields where numbers and text will be displayed; text fields for user input and buttons to initiate actions.
2. Right-click and connect all labels by drawing a line from the Storyboard to the ViewController. Assign variable names to all labels and text fields, and function names to all buttons.
3. In the ViewController, declare variables that will be used to hold data entered into text fields and calculations made by functions. Variables can be one of four types—string, integer, floating point, or double precision (**Table 16-2**). Each variable needs to have its type declared and be set to zero or blank.

Table 16-2. Variable Types in Xcode

Variable Type	Abbreviation	Description
String	String	text and numbers
Integer	Int	whole numbers (no decimal)
Floating Point	Float	numbers with decimal places, 32-bit accuracy
Double Precision	Double	numbers requiring high precision, 64-bit accuracy (usually not necessary for most apps)

Write Code

1. Inside the function, write code to store the text field numbers entered by readers into variables of type

"String." Note that the text fields are converted to integer variables by adding the type Int to the front of the statement. Also both variables need to be "unwrapped" using an exclamation point.

```
numberOfApples = Int(applesTextField.text!)!

numberOfOranges = Int(orangesTextField.text!)!
```

2. Write a statement to add the numbers of apples and oranges:

```
numberOfFruits = numberOfApples + numberOfOranges
```

3. Write a statement to output the number of fruits to the label field:

```
answerLabel.text = String(numberOfFruits)
```

Again the variable type must be converted from integer to string before being written to a label because labels can contain only string variables.

Debug and Test

1. Xcode has a live debugger. If the program finds errors in the code, it shows a red stop sign icon with an explanation. All errors must be fixed before the program can run.
2. Hit the Run button in the upper left corner of the screen to launch the Simulator and run the program. If the program has errors that have not been detected by the live debugger, the program could crash. A common source of crashes is multiple connections from one Storyboard object to the ViewController. In this case check the Connections Inspector (right-facing arrow at the top right of the screen) to make sure each object has only one connection.

Create an Icon

Icons for iOS apps can be created in Adobe Illustrator, Photoshop, and similar programs and saved in PNG format. The authors suggest preparing a document at 1024×1024px, the maximum resolution required.

An icon design should be very simple and not include a lot of text, which could be difficult to read on a small icon. Some icons consist only of a logo or one or a few letters.

An app requires several images of various resolutions for different Apple devices, screen sizes, and locations. Numerous "app icon maker" sites exist that will accept a 1024×1024-px PNG file and create the various resolutions required.

Install

1. Programmers who sign up for the free version of the Apple Developer Program can install apps on an iPhone or iPad connected to the Macintosh computer via a USB cable. However the app will expire

after 7 days.

2. Users who sign up for the paid version of the developer program can install apps as above with no expiration date and also distribute them to beta testers.

Making a Web Portal App

Many apps are actually "web portals" that lead to a web site, where users can get information, search, buy things, and perform other tasks. A web portal is easy to make if you have an existing web site that is styled using responsive design and is easy to read on a smart phone. Requirements for a web portal app include:

- The author should have access to a web site with responsive design.
- The web site should accept secure login (https).
- A web portal can be written with only two lines of code. (Readers are referred to Apple documentation or one of the many available videos for the most up-to-date code.)
- In the info.plist file, the programmer should set the App Transport Security Settings > Allow Arbitrary Loads in Web Content (Boolean variable) to "YES."

For Further Information

Readers are referred to the many resources available from the Apple Developer Program, Apple's free iBook *App Development with Swift,* and developers who post videos on YouTube.

Chapter 17 - References

W3 Schools • Web Reference

W3 Schools is a handy online reference to HTML tags, CSS styles, special characters, and other information. If you're not sure how to write a tag or a style or are having difficulty getting it to work, look it up on W3 Schools using the Search function.

Balsamiq • Mockup Tool

Balsamiq is an online mockup tool for designing web pages.

jQuery • JavaScript Library

jQuery is a JavaScript library of functions that aims to simplify programming and adding special effects and interactivity to web pages.

jQueryUI.com contains code to further extend jQuery with visual support and many functions such as date-pickers, sliders, and drop-down menus.

Vue.js • JavaScript Framework

vuejs.org provides a wide variety of JavaScript based functions to augment the interactivity of a website/

"CSS Diner" • Practice with CSS Selectors

The CSS Diner site is a 26-level interactive exercise for practicing CSS selectors.

CodeAcademy.com • Web Reference and HTML Tutorial

CodeAcademy.com offers tutorials on HTML, CSS, JavaScript, and other topics related to the web.

Apple Xcode for iOS Apps

Everyone Can Code: App Development with Swift

Image Credits

Chapter 1 – Introduction

Figures 1-1, 1-2. “Adobe® Dreamweaver® screenshot(s) reprinted with permission from Adobe Systems Incorporated. Adobe® Photoshop® are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. Screen captures cannot be cropped further.”

Chapter 2 – Files and Links

Figures 2-1, 2-2, 2-3. “Adobe® Dreamweaver® screenshot(s) reprinted with permission from Adobe Systems Incorporated. Screen captures cannot be cropped further.”

Chapter 4 – The Semantic Web

Figure 4-1 (left). “Adobe Dreamweaver screenshot(s) reprinted with permission from Adobe Systems Incorporated. Adobe® and Dreamweaver® are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. Screen captures cannot be cropped further.”

Chapter 5 – Styles and CSS

Figure 5-1. “Adobe Dreamweaver screenshot(s) reprinted with permission from Adobe Systems Incorporated. Adobe® and Dreamweaver® are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. Screen captures cannot be cropped further.”

Chapter 6 – Image Optimization for the Web

Figure 6-4. “Adobe Photoshop screenshot(s) reprinted with permission from Adobe Systems Incorporated. Adobe® and Photoshop® are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. Screen captures cannot be cropped further.”

Chapter 8 – Animation

Figure 8-1, 8-2. Tumult Hype screenshots reprinted with permission from Tumult Inc. Figure 8-3. “Adobe Dreamweaver screenshot(s) reprinted with permission from Adobe Systems Incorporated. Adobe® and Dreamweaver® are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. Screen captures cannot be cropped further.”

Chapter 12 – eBook Formats

Figure 12-2. “Adobe InDesign screenshot(s) reprinted with permission from Adobe Systems Incorporated. Adobe® and InDesign® are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. Screen captures cannot be cropped further.”

Chapter 13 – eBook Production

Figure 13-2. “Adobe InDesign screenshot(s) reprinted with permission from Adobe Systems Incorporated. Adobe® and InDesign® are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. Screen captures cannot be cropped further.”

Chapter 14 – Digital Photography for Web and Cross-media

Figure 14-16. “Adobe Digital Negative Converter screenshot reprinted with permission from Adobe Systems Incorporated. Adobe® is either a registered trademark or trademark of Adobe Systems Incorporated in the United States and/or other countries. Screen captures cannot be cropped further.”

Chapter 15 – Photoshop for Web and Cross-media

Figure 15-1, 15-3, 15-4. “Adobe Photoshop screenshot(s) reprinted with permission from Adobe Systems Incorporated. Adobe® Photoshop® are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. Screen captures cannot be cropped further.”