Boot Camp
Design Essentials for Beginners

Web Authoring Boot Camp

Get started as a kick-butt web author with must-have info on web project planning, workflow, and code

L.J. Bothell

Web Authoring Boot Camp

kick-butt web authoring

L.J. Bothell

studio bast
Seattle, WA
www.studiobast.com

Contents

Web Authoring **Boot Camp**

WHAT’S GOING ON?

section 1

Web authoring bootcamp, that’s what!

In this section, we’re going to find out what web authoring is, and how planning, preparation, and research can do to help you design effective and usable websites.

intro
Introduction to Web Authoring Boot Camp

Well, lookee here, it's another web design book. As if there aren't already enough of them in the bookstore and sold through Amazon. What can this book possibly offer that I want to know?

Why Another Web Design Book?

It's true, there are tons of web design, coding, authoring, production, styling, programming, analyzing, and ice-cream sundae-making books on the market. Many of them are really good, too, like anything from PeachPit Press, Sitepoint, and Sam's Teach Yourself. Get one or two of these – a how-to and a reference-oriented book, and you'll have a terrific start to educating yourself on the wonderful world of creating websites and web media. Excellent!

This book is a little different, for several reasons:

- It's free.
- It's open-source, so that it can be openly distributed and used for non-commercial education purposes.
- It's the first put-together open-source book available on web authoring (as of this writing).
- It covers not only the how-tos of building websites, but also treats the whole website project – from planning to completion.
- It looks at web authoring as a profession, a workflow, and a great career to get educated in.
- It's written by a web author who has education and experience in the field and who also gets to teach this information to students.

Who This Book is For

This book is for new web authors – those who are taking classes, teaching themselves through tutorials and videos, and anyone who wants a glimpse into the world of web authoring – as a profession as well as a workflow practice. It's for you if:

- You want to know what a web author is and does for a living.
- You want to understand the web project planning and preparation steps that happen before coding.

- You want to build usable, accessible, and intuitive HTML websites that anyone can use.
- You want to get step-by-step basics on coding practices and workflow.
- You want to launch well-designed, W3C standards-compliant websites with basic SEO practices.
- You are starting/taking classes in web design/production as career training.
- You want to build your own websites, freelance build for other clients, and/or work with creative talent agencies and employers who hire designers, authors, producers, and other web specialists.

This book is not really for seasoned web authors, although it's a terrific reference tool after you have used it to get your feet wet. Is covers the essentials of planning, preparation, designing, building, and launching of static HTML websites, and refers readers to online sources and more detailed books for advanced web authoring like stylish CSS, HTML5 and CSS3, Javascripting, and other web authoring languages and scripts. It's not for you if:

- You already have a several websites of experience under your belt.
- You already work in the web specialties field and are past beginner-stage.
- You are looking for dozens of specialized tricks, styles, and recommendations for intermediate/advanced CSS.
- You want to build dynamic websites that are server-side and database driven.
- You want detailed workflow and training on HTML5 or CSS3.

How to Use This Book

This book is pretty compact, and is more-or-less a process/workflow book. It combines a conversational tone with explanations, details, examples, and factoids. The goal is to give you what you really need to get started as a kick-butt web author, rather than every detail known to web geek – that's what the Web is for. Occasionally, I'll make a joke or ask you a question, and I expect you to chuckle and to answer. Heh – gotcha!

Each chapter introduces itself, then gets into the nitty gritty of the subject, requirements, tools, and workflow. You are offered examples of code and illustrations, plus occasional exercises to try out yourself. Have a willingness to try what you see and to explore more through other examples and tutorials on the Web.

You can start at the beginning to learn what web authoring is and how to plan web projects in Section 1. You can jump right into basic HTML coding and styling in Section 2.

You can get started creating a web page template or a basic Cascading Style Sheet. You can use the chapters and resources at the end to jumpstart problem-solving or adding bits of dynamic stuff to your sites.

This book also makes a pretty handy virtual paperweight.

Learning Tips

- Take your time. This book is meant to be used, not just read. Look at and try out the code.
- Slow down when you need to so that you don't overload your brain. Focus on a chapter section or three, take a breather, then go back and make sure what you are learning makes sense and that you can use it.
- Take the pieces of code you see and work to assemble them throughout the book as you learn the elements and workflow for HTML coding.
- Share what you learn. Talking about it, and showing someone else what you are planning and doing helps make stuff stick.
- Make sure to stay hydrated and have chocolate somewhere nearby. Learning sticks better with chocolate. Everything is better with chocolate.
- Create something! This book will help you best when you come up with a simple website idea of your own and apply the planning, preparation, design, coding, and validating steps for yourself.

1 The Web, The Code, and Everything

What is the World Wide Web?

The World Wide Web (commonly abbreviated as the "Web") is a system of interlinked hypertext documents accessed via the Internet. With a Web browser, you can view web pages that contain text, images, videos, and other media, and navigate between them using hyperlinks.

History

The World Wide Web Consortium (W3C) is the main international standards organization for the Web. This consortium develops and upgrades the standards for the web. If you don't like something, it's their fault.

Seriously, though, Tim Berners-Lee published what is considered to be the first website in August 1991. Berners-Lee was the first to combine Internet communication (which had been carrying email and the Usenet for decades) with hypertext (which had also been around for decades, but limited to browsing information stored on a single computer, such as interactive CD-ROM design).

Then

Originally meant for the free sharing of knowledge, the Web has transitioned from basic documents to the use of URLs to get to addresses, and from simple text and basic image pages to highly styled websites. It's gone from scientific and information-based to edutainment, e-commerce and promotion-based sites. It has become a primary way for us to get research, news, products, social interaction, etc.

It wasn't always this way. Back in the day, websites were written in a markup language called HTML, and early versions were very basic, only giving a website's basic structure (headings and paragraphs), and the ability to link using hypertext. This was new and different from the then existing forms of communication, since visitors couldnow easily navigate to other pages by following hyperlinks from page to page.

Now

Right now the Web is becoming our primary source of information, entertainment, education, shopping, promotion, and more. There are applications, social media, multimedia, games, and just about anything you can imagine. If it's not there, it soon will be.

Basically, the Web is a worldwide storage basket of all things info, and likely more aggregation, easier access, more personalization, more "plugging in" etc. will happen.

HTML

HTML (Hypertext Markup Language) was the original simple way to place text and images in a web page, and browsers were created to recognize the language and render it for viewing. It is based on a series of tags, which could be used without closing or doctype declarations, and simple inline text formatting styles like bold and italic. It allowed for fairly simple report-style information formatting.

As the Web and browsers progressed, more visitors gained access to information and later product marketing. They needed attractive, more easy to read web pages. Web design progressed, and the markup language became more complex and flexible, giving the ability to add objects like images and tables to a page. Features like tables, which were originally intended to display tabular information, were soon subverted for use as invisible layout structures.

The potential for marketing, entertainment, and profit on the Web pushed the boundaries of simple HTML far past where its original standards were meant to go. Visitors to web pages wanted to see more excitement, so HTML styling commenced, with inline styling of widths, heights, text formatting, image and link modifications, and more. Plus, as browsers developed, web designers created hacks and workarounds to get HTML to do their bidding. Which, it turned out, different browsers and monitor resolutions interpreted differently. A web page on one browser, like Internet Explorer, could look pretty good, while on Firefox or Safari it would look equally poor. Or vice versa.

In the 1980's and 90's, Cascading Style Sheets (CSS) were developed to extend HTML styling and allow additional standardizing information to be developed to render more complex pages. The first officially adopted version was available in 1998. Web pages could now have complex styles created and stored in a separate file, which, when updated, would cascade those changes throughout all pages on the site that referenced the CSS file. The styles could look a lot more similar between browsers, which were also becoming more standardized as new versions came out. However, as marketing, sales, entertainment, and visitor needs for attractive, dynamic, and increasingly robust web content display grew, even more had to be packed on top of HTML and CSS.

Website Building Enhancements

For more interactive-seeming pages, Flash and JavaScript have been added to the mix.

They allow for style enhancements like galleries, slideshows, banner ads, rollover navigation, and more. These make pages look alive and refreshed, but don't usually allow real interaction between the visitor and the page/page creator. These, and some AJAX, are client-side tools (you and the browser being the client, or receiver).

Client Side

Client-side scripting refers to programs on the Web that are executed by the visitor's web browser, instead of on the website owner's web server. This dynamic HTML (DHTML) enables web pages to be scripted; that is, to have different and changing content depending on visitor input, environmental conditions (such as the time of day), or other variables.

Web authors write client-side scripts in languages such as JavaScript, which are often embedded within an HTML document (embedded script), They may also be contained in a separate file, like JQuery, which is referenced by the document (or documents) that use it (external script) by links in the web page's header section.

Server Side

Server-side scripts are executed by the web server when a visitor requests something (like a login approval). They produce output in a format understandable by web browsers (usually HTML), which is then sent to the website visitor's computer. The visitor cannot see the script's source code (unless the author publishes the code separately), and may not even be aware that a script was executed. Documents produced by server-side scripts may, in turn, contain client-side scripts. The visitor's request is then fulfilled by running a script directly on the web server to generate dynamic web pages. It is usually used to provide interactive websites that interface to databases or other data stores.

Server side tools/languages have made web pages almost completely dynamic/interactive. Languages like ASP, Python, PHP, Java, C#, and Visual Basic work with information databases to build and drive dynamic websites. The visitor can input information to get specific pages rendered especially for him/her, and to store data so purchases can be made, etc. Look 'em up online to learn more.

XHTML

A lot of websites do not need to be fully dynamic, and HTML limitations for non-dynamic sites have long since been blown. Intermediate web production is all about moving past HTML to the more precise, manageable, and World Wide Consortium (W3C) validated current standards for basic web coding. This means that with increased standardization, most browsers show the same results from code, and the code has been expanded to allow for more general style and basic tag use.

Original HTML coding was very forgiving (it didn't require a doctype, charset, namespace, closing tags, or avoidance of deprecated tags to still show the web page). HTML does require more tweaking as a website grows to work with multiple browsers and quirks. This makes web production unwieldy and not very cost or resource-effective.

As a result, in February 1999 XHTML 1.0: The Extensible HyperText Markup Language was developed. In January 2000 it was officially adopted as a W3C recommendation. It has been designed by the W3C to merge the best of HTML and XML (Extensible Markup Language), to work fluidly with the easy-to-organize CSS, and to be able to be parsed by every browser with as much uniformity in interpretation as possible. CSS had already been adopted for enforcement and validation in 1996.

XHTML is stricter than HTML, because it absolutely requires:

- A doctype declaration.
- Use of HTML, head, and body tags.
- All tags to be closed.
- Use of quotes around values.
- Use of all lower case elements, attributes, and predefined values.
- Use of the ALT attribute and value for all images.
- Avoidance of deprecated tags that are being phased out by the W3C.

While this can sound pretty tough, it actually is easy to learn. It also results in organized and consistent code, allows clean validation, and keeps you away from deprecated tags and quirky practices that give browsers (and you) huge headaches. HTML5, which is currently being implemented but which is not well supported by many browsers yet, keeps these expectations intact.

Web Addresses

Anyone who has searched for something on the Internet using Google or another search engine has seen the results, with hyperlinks to the related website. The address of these websites are known as URLs. The URL (Uniform Resource Locator) is the full web address: **http://www.studiobast.com.**

HTTP (Hypertext Transfer Protocol) is a protocol for distributed, collaborative, hypermedia information systems. It is used for retrieving inter-linked resources, and is integral to the universal web address: **http**://www.studiobast.com. Sometimes you'll see HTTPS, which is a secure version of the URL, such as when you have logged into a website. This tells you that you are looking at/working with hyperlinked pages.

WWW is the part of the URL address that reaches out to the World Wide Web: http://**www**.studiobast.com.

The **Domain** name refers to your domain, or the domain the Web host you use has assigned to you: http://www.**studiobast**.com, http://www.**freehostia**.yoursite.com.

The **Subdomain** name is your contribution to the URL of a site you have hosted on someone else's web host domain: http://www.freehostia.**yoursite**.com.

The **Host** name refers to where the website is hosted: http://www.studiobast.**com**, where .**com** indicates this is a commercial address. You will also see .net, .info, .edu, .gov, .ca (for canada), etc.

Web Work

Web pages have gone from a few dozen to millions, with more exploding daily. Many of these are part of larger websites – the collection of web pages making up a theme and collection of information. A whole industry in Internet Technology is made up of those who plan, design, produce, populate, test, and distribute web pages/sites. What have we really got out there in the Web specialties, besides increasingly high-fallutin' titles?

Front-end

The observable content (like page layout, visitor interface, graphics, text, audio) is known as the front-end.

The front-end of web work includes the visitor-facing stuff: the web pages, forms, graphics, general interaction, content, and so on. These things all take different skills, and people can specialize in, or mix it up in, several areas:

- **Graphic Design/Concept**: The layout/look/palette/type and CSS of websites.
- **Visitor Interface/Experience Design**: Design based on the focus of the visitor's experience and interaction, including experience testing, pre design surveys and interviews, focus group creation, and all things that make the visitor's interaction as simple and efficient as possible.

- **Interactive Design**: JavaScript, Flash, streaming media, and other multimedia and scripting client-side expertise.
- **Prototyping**: The build-out of the web concepts in a working prototype before full coding and programming happens. In designer/developer teams, this is considered the pre-build.
- **Content Management**: Providing, placing, and/or maintaining the information that makes up the content of the website, like text, images, links, animations, etc.
- **Usability**: The practice of making sure a website is accessible, has usable visitor interaction (like navigation), accessibility with other mediums like print and mobile devices, and search engine optimization.
- **Production**: The update and maintenance of websites, such as new pages, code cleanup, adding and removing content, and checking links.
- **Marketing/Promotion**: Email marketing, site listing and promotion, SEO, etc.
- **Webmaster**: Multi-skilled individuals who carries out many of/all of these tasks for a whole website.

Back-end

The back-end comprises the organization and efficiency of a website's source code, invisible scripted functions, and the server-side components that process the output from the website visitors. This requires the more programming-oriented work that makes complex interactive websites work. Does not usually include any visual design, but rather has designers and other front-end folks as part of the collaborative team. Back-end work includes:

- **Developer/Programmer**: Main non-design aspects of building websites, such as writing markup and language coding. Web development can range from developing the simplest static single page of plain text to the most complex web-based internet applications, electronic businesses, or social network services.
- **Database Administrator**: A database administrator (DBA) is a person responsible for the design, implementation, maintenance and repair of an organization's database, which drives fully interactive sites like Amazon. The role includes the development and design of database strategies, monitoring and improving database performance and capacity, and planning for future expansion requirements.
- **Analyst**: Web analytics is the measurement, collection, analysis and reporting of internet data for purposes of understanding and optimizing web usage. This provides data on the number of visitors, page views, etc to gauge the traffic and popularity trends which helps doing market research.

- **Security**: Web development takes into account many security considerations, such as data entry error checking through forms, filtering output, encryption, and security patches.
- **Project Manager**: Oversees collaborative design between group members with specialized skills.

What is Web Authoring?

For the purposes of this book, we are going to describe **web authoring** as the work involved from the beginning to the end of a web project. Web authors don't just code/build cool sites; the really employable and valued web specialists train and gain experience in the whole range of skills that are involved in designing the website concept, producing it, and testing/releasing the website for launch.

Planning

A web author has to start at the beginning, even if s/he inherits a previous web project. This includes getting to know the website client and the requirements of the project, often through client meetings, and/or a client survey. You need to set up your administrative information for organizing and billing the project, like contracts, charter, forms, communication methods, etc. Planning also includes getting clear on who the website visitors are going to be, and what they are supposed to accomplish/ Web aothors use planning tools like an AIA table, wireframes, project calendars, interviews, and various research methods. Finally, planning includes setting the timeline and budget, researching the website needs, and gathering the resources and tools needed to do the work, like the content, images, web host, etc. You can't design a website if you don't know why and for whom.

Design

The design part of web authoring goes hand-in-hand with understanding the expected website visitors, accessibility, SEO, and usable layout. Design isn't just about being cool, but is mostly about crafting a website design that the visitors can actually easily use and react to. This also includes doing the anticipated sitemap, wireframes, and approval comps. You can't produce a website until you have a clear, and client-approved design plan.

Coding

Coding the website is part of the production, and means writing the actual HTML and CSS (and related client-side script) of the website. Some web builders like to use WYSI-

WYG tools for this, where they click on icons and assemble pieces in a visual way, rather than writing the code itself. Some programs are very good with being clean code translators, while others are not.

Production

Production refers to the assembling of the website, including the coding, placing of images and media, linking, and creating clean navigation elements. It can also include the creation and/or editing of content, like received text or stuff you write, preparing images for web readiness, updating existing pages, and merging with tools and resources that the website might provide to visitors.

Pre-launch Preparation

Before you can launch a website, you have a whole bunch of considerations on your checklist: revisions, finding/contacting a web host, revisions, uploading, page testing, client-side testing, server-side testing, form testing, link testing, cross-browser comparisons, revisions, validation, revisions, and more revisions. Coffee??

Launch/Maintenance

Whew! Then you are done. Or are you? Part of your relationship with your client/web project is getting it launched and closing the project properly. Launching includes making sure the site will be indexed by search engines and that it is reachable on the Web by a URL and a visitors' search. Closing the project means knowing who will maintain the site (if you will not) and making sure you leave clean and accessible code for your replacement, as well as any necessary documentation about quirks and concerns. Finally, you need to close your administrative functions on the web project: invoice, turning over files to the client, etc.

Coding vs. WYSIWYG

Code Authoring

As mentioned above, code authoring is where you hand-code in HTML, CSS, JavaScript, and any other languages or scripts you choose. You have complete control over the output of the website when you code from scratch. WYSIWYG programs can reinterpret your visual design choices with poor code that won't be up to current W3C standards and won't validate or look right on necessary browsers. When you know your code you can correct anything, create your own templates and snippets, and generally offer more professional value to employers and clients.

WYSIWYG Building

WYSIWYG is an acronym for *What You See Is What You Get*. The term is used in computing to describe a system in which content displayed during editing appears very similar to the final output. It allows people to edit web pages without knowing HTML. This is good on the fly for quick edits, but if you plan to do this as a regular web authoring practice, you won't be a serious contender for the web specialties because employers and clients often do not have or allow use of WYSIWYG programs for their professional websites.

This book will focus on the basics, next steps, and some special tips for web authors who will hand-code. You will learn how to accurately and cleanly code original web pages, create streamlined workflow process, and leave happy website clients who will want you back for more. That's the goal, anyway. Let's get going!

2 Web Projects

First, understand that a website is a collection of web pages that hyperlink to each other, use a consistent theme, strive to serve a basic purpose, and offer cohesive content that communicates to website visitors. It could be YouTube, Amazon, your local used bookstore's website, your blog, the local community college, etc. These sites share one thing in common – they are a collection of hyperlinked pages that are mostly related to each other in subject coverage and theming.

Axiom #1: A website is a method to communicate key information to and prompt action from an audience, and all design, content, and navigation options must serve this axiom.

How do you get from here (total noob and/or developing web author) to there? We have a LOT to consider. Check it out!

Website Planning

What do you need to consider in developing a website plan? For typical website construction, the basic aspects you need to consider include:

- **The usability**: The site should be visitor-friendly, with the interface and navigation made simple and reliable.
- **The content:** The substance and information on the site should be relevant to the site and should target the audience(s) that the website is focused upon.
- **The appearance**: The graphics and text should include a consistent and clean style that flows throughout the website. The style should be professional, appealing and relevant, primarily to the target audience.
- **The visibility**: The site must also be easy to find with major search engines and advertising media.

Usability

Imagine a website where the visitor knows exactly what page s/he is on (and why), can navigate really easily and fast to other pages, and gets exactly the information / completes exactly the right task with absolutely no fuss. Sound easy? It can be.

Focusing immediately on web usability is an approach that makes websites easy to use for a visitor, without requiring them to have any specialized training. The visitor should