In this chapter, I will introduce accessibility, define what it is, and discuss why you should be paying attention to reducing the barriers to access that might exist for many people in the systems you build. We will also look at HTML5 and examine its main differences from HTML 4—and you’ll see what this means for accessibility as well as how those differences will impact the way you build web sites and applications.

Introduction to HTML5: The New Wave

HTML5 is here. It is the new lingua franca for the Web. So what is it? HTML5 is the new version of HTML 4, XHTML1, and DOM Level 2 HTML. It has moved HTML from being a relatively simple document markup language to being a sophisticated platform for web applications with a host of new, rich application programming interfaces (APIs).

As with all major changes in life, the transition might not be smooth. Such a major shift that adds a host of new elements and attributes presents particular challenges for you as an author as well as, potentially, for the consumer.

HTML 4 was readily understood and offered features that could be used in imaginative ways. By leveraging these features in combination with other languages like Cascading Style Sheets (CSS) and JavaScript, developers could do things that were increasingly complex, pushing the models of user interaction.

With the advent of Web 2.0–type content such as AJAX, dynamic content updating, and more client-side processing, we saw a variety of terrific, sophisticated applications being developed out of what were often semantically limited markup languages.

This new wave of glossy widgets—and sometimes bizarre interaction models—often present many challenges to the user. If the user has a disability, the challenge is greater, because that user might not be able to access important widget functionality from the keyboard or content updates might be lost on the screen-reader user. Long before HTML5, many web designers started to really care about accessibility and look for ways to ensure their web sites and applications were usable by the widest audience. So grassroots movements like the Web Standards Project were born. An active and vibrant community arose that saw the challenges of improving the user experience and quality of design as a call to arms.

If you are reading this, chances are you are a web designer or developer and you would like to know more about HTML5 and how it relates to accessibility. So you’re in the right place. This book assumes you have a decent level of knowledge of HTML 4, as well as some CSS experience and maybe some JavaScript experience. It also assumes you’re keen to learn about developing robust web sites and applications by using HTML5.

If you’re not an expert CSS or JavaScript person and don’t even know much HTML, don’t worry! This book is designed to provide sufficient references to online materials and resources that will soon get...
you up to speed. The HTML5 spec is notoriously large (about 800 pages), so what this book tries to do is distill the parts that relate most to developing accessible web sites. This book (in tandem with some other more generic HTML5 resources) should help you come to terms with the game-changer that is HTML5, as well as accessibility.

In this book, I will attempt to share with you what I know about both—as a web designer and developer and as someone who has worked with people with disabilities for nearly 10 years. I am also a member of the HTML5 working group, where my input has had particular emphasis on accessibility and on trying to ensure (with many others) that the HTML5 specification will best serve the needs of the broadest range of users, including people with disabilities.

HTML5 vs. HTML 4

So what’s new in HTML5, and how does it differ from HTML 4? First, HTML5 is designed to do an awful lot more than just mark up text or be a hypertext markup language. (You are forgiven if you thought it was.) Second, many new APIs are now contained within the new specification that might not seem, at first glance, to belong in a document markup language at all.

These APIs are many and varied, and include Web Workers (an API for running scripts in the background independently of any user interface scripts), Web Storage (similar to HTTP session cookies, for storing structured data on the client side), and Web Sockets (for bidirectional communications with server-side processes). When you factor in native, “in the browser” support for video and audio via the <video> and <audio> elements—which signal a significant move away from browser plugin solutions like Flash, as well as the new 2D drawing API <canvas>—it’s obvious that HTML5 is far more than a mere document markup language and really is a quantum leap beyond the other earlier versions of HTML.

**Note** The added features are a mixed blessing. With these new language features, there will be more functionality but also more complexity both for you as an author and for your users. However, employing best practices in terms of user interface design and also in how you approach coding your projects will stand you in good stead. So while it might feel a little like you have to start all over again, I hope you will see that it doesn’t entirely have to be like that. You will just have more choice and, of course, learn a few things. I hope also that the good things you might know about making stuff accessible using a combination of HTML 4 and CSS/JavaScript will often still apply. Don’t be worried about keeping things simple—just using the right tool for the right job can get you very far.

HTML5 Syntax

In terms of syntax, HTML5 is a language that wears two hats. It can be written as both HTML and XML (also known as XHTML, which is an XML serialization of HTML—or put another way, an XML-like version of HTML that brings some of the rules of XML syntax to HTML).

Depending on your own requirements, you might need to serve more strictly well-formed XML-type documents to your users. Doing that will require the use of an XML parser, which is used to process XML documents. Alternatively, content comprised of HTML that is syntactically more lax HTML might suffice for your needs, in which case you can use an HTML parser. “More lax HTML” refers to code that might be a little sloppy but still works.