Tools

It's hard to overestimate the value of tools for programming languages. There are tools for creation, testing, debugging and even measuring the performance of a program. HTML pages are really programs, in specific programming languages, so you need tools for those as well.

One tool that you should, by now, be very familiar with is the editor. While traditional text editors allowed you to enter and edit text, modern editors can actually help you detect and avoid errors, visualize your program more clearly, and even stick in bits of text for you if it thinks it knows what you might need, e.g. close tags.

Some of the most ubiquitous tools are in the browser itself, which is handy because everyone seems to have one. You might have even seen this yourself if you've ever tried the "Show Source" button, but perhaps the most important tool that we have yet to explore is the debugger activated when you "Inspect Element".

ACCESSIBILITY

While it's important that your page looks and acts the way you intend, there are other considerations as well. One important aspect is "accessibility". When talking about Web pages, accessibility means designing your page so with various disabilities in mind. For someone with impaired vision, you'd want to make sure that your page is zoomable, or make sure that it makes it easy for screen-readers. Just as wheelchair ramps are a benefit for many users (ask anyone who's had to drag a stroller up the stairs), making your Web site accessible can be a benefit for many users.

Fortunately there are a number of tools available to help evaluate the accessibility of your site. You can evaluate the overall accessibility of your page, see how it looks to a screenreader and even figure out how well your background a foreground colors interact. For a good list of such tools, check out Web Accessibility Evaluations Tools List.

DEBUGGING

When you're using a traditional programming language, like JavaScript, debugging is primarily concerned with values of variables, what path is being taken through the code, and whether the program crashes, usually resulting in everything after the crash not happening at all. JavaScript is what you could call a "Procedural" language, in that things follow a procedure in order described by the program. Just think of how you might direct a small child, "Go into the bathroom, stand in front of the sink, pick up your toothbrush, then put toothpaste on it and put the toothbrush in your mouth, but don't swallow the toothpaste etc, etc." It can be a bit tedious, but then you've got a better chance that the end result will be accomplished.

HTML and CSS are not procedural languages, but rather are "Declarative". You declare what you want, and the computer makes it happen. You say "this is the header, here's a paragraph, I want this to be large, that to be red, on your mark, get set GO!". You don't proscribe exactly how it's done, just want you want to the result to be. Now you're dealing with something more like a teenager - "I want to see this room sparkle! And I should be able to bounce quarters on that bed!". Less tedious, but sometimes it's a bit tricky to get exactly what you're hoping for.

In either case, you might need some help. With a small child, you might rely on an older sibling reporting exactly what happened, so you can correct any unexpected variations that might occur. For teenagers you can usually listen at some distance and figure out what's happening (or not happening).

Since we are focusing on HTML and CSS, let's consider what sort of info may be helpful. There are several things that can go wrong with our programs. Some text might be missing. Some things will be in the wrong place. Some things might be too close together, while others are too far apart. You thought you had everything right, but it doesn't look quite right. How to figure out where the problem is? That's where Development tools come in.

Every browser is a bit different, but most of them have ways to examine the various elements and their properties. It probably began as a way to see what's going on with particular elements of the HTML page, at least or so it seems as "Inspect Element" seems to be common to most browsers. We'll be using the debugging facilities in the Intel® XDK, but most of what we'll cover uses the "Chrome Developers Tools" which is the same tool you get when you "Inspect" an element on Chrome (or Chromium). Other browsers are likely to have similar capabilities, but the user interface may be a bit different. Feel free to use whichever tools you prefer, but the examples will be easiest to follow if you use Chromium, Chrome or Intel XDK.