Chapter 4: What Is CSS?

What Is CSS?

All modern forms of publishing (not just Web development) follow a principle of *separation of structure and presentation* (or *separation of content and presentation*). In this context, *content* and *structure* have to do with the actual words and design elements on the page. For example, these words you're reading right now are *content*. The structure has to do with the various kinds of elements you see on this page, such as headings, paragraphs, lists, and pictures. The content and structure are primarily handled by HTML.

The term *presentation* has more to do with design and aesthetics—things like colors, background colors, fonts, and other things that developers apply to the structure and content to give it its style—its look-and-feel. In modern Web design and electronic publishing, we use CSS (*Cascading Style Sheets*), a separate language, to define those things.

Obviously, when you look at a page like this one, or any other published page in any website, magazine, or book, the content, structure, and design all look right on the page. They're not visually separate for the person viewing the page. But they are separate for the person *creating* the page because there's a different language for each. To the *user* (or *reader*, or *consumer)*, the person who's just looking at the final product, there is no separation. But to you as the *developer* (or *creator*, or *designer*, or whatever you want to call yourself) there is separation in terms of *how* you create; that's why there are two separate languages. In simplest terms:

Text equivalent start.

HTML for structure and content (what each element is). CSS for presentation (how each element looks)

Text equivalent stop.

To the novice, this might seem frightfully vague, and maybe it even seems like a mistake that overcomplicates things. Maybe if you just wait long enough, the people who define these things will come to their senses and come up with some simple single language that does it all. Well, if you're thinking that, don't hold your breath. As it turns out, just the opposite is true. At first, and for several years, there was only HTML (no CSS), and it proved to be a nightmare for large-scale Web development. CSS came later to solve the many problems of trying to do everything with just one language. Nobody will ever want to un-solve the many problems that CSS solves.

The idea of having two languages was nothing new at the time. People in print publishing, desktop publishing, and word processing have been using *style sheets* successfully for years, which is a method for separating structure from presentation. The invention of CSS was just a matter of taking what was already successful in other forms of publishing and applying it to Web development. And its great success in Web development is what's driving its spread into other forms of electronic publishing and application development.

Like HTML, CSS is a language that's defined by the World Wide Web Consortium. And like all languages and software products, CSS has also evolved through multiple versions over the years:

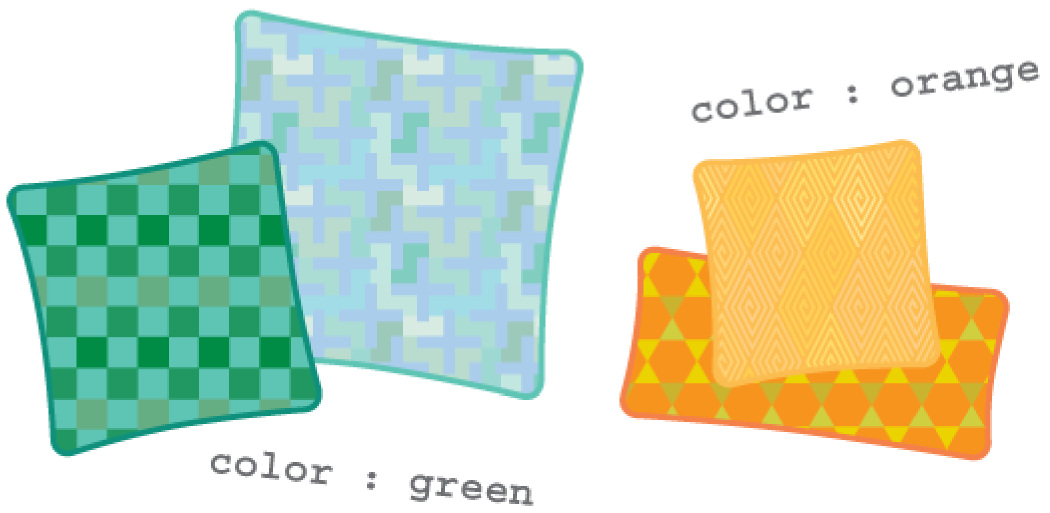
Version	Year
CSS1	1996
CSS2	1998
CSS2.1	2011
CSS3	Candidate

CSS Versions

The current recommended CSS version is CSS2.1. CSS3, like HTML5, is newer and hasn't yet reached that full recommended status. But it's far enough along now that few (if any) changes are likely to take place. Web browsers and other user agents are well on their way to supporting its features. Many developers are currently using it for websites and other types of documents, and more will be using it in the future. Now is the time to start learning CSS3, and so that's the version we'll be discussing in this course.

Properties and Values

While HTML involves using tags and attributes to define what each element *is*, CSS involves using *properties* and *values* to define how each element *looks*. The term *property* comes from the way we use the term in the everyday world. For example, think of any object. Say, a pillow. There are lots of pillows, but what makes one pillow different from other pillows? Its unique properties. For example, every pillow has a height, width, and color. Things like *height*, *width*, and *color* are properties in the real world, and they're also properties in the CSS language. The various properties that make one particular pillow different from others are sometimes referred to as its *style*. We use properties in CSS to define the style of things on the Web pages we create.



You can think of the property as the *what-to-style* part (height, width, color). Think of the value as the *how-to-style* part. For example, *color:red* means "make the color (property) red (value)".

The CSS syntax requires that the property name always comes first, followed by a colon, followed by the value you're assigning to the property, like this:

property:*value*

Again, using our simple color example, *color:* is an actual CSS property that refers to the color of text. You can apply a color value to it. For example, *red* is a valid color name value in CSS. So to make the text red, you type the property:value pair like this:

```
color:red
```

You can put a space before the colon, after the colon, or both. It doesn't really matter because spaces there are optional and the code works the same with or without spaces. Sometimes you may see a property:value pair typed like this:

```
color: red
```

Or even like this:

```
color : red
```

Nothing to worry about, they're both fine and work exactly the same.

You can apply multiple *property*:*value* pairs to the things you're styling. But when you use multiple property:value pairs, you must use a semicolon (;) to separate each pair, like this:

property:*value*;

property:*value*;

property:*value*;

The semicolon after the last property:value pair is optional. There's no harm putting it there, and there's no harm leaving it off. When you're looking at other peoples' code, you'll often see that the last property:value pair has a semicolon at the end. The main reason developers do that is so that if later they decide to add another property:value pair to the list, they won't forget to type the necessary semicolon at the end of the property:value pair above the new one they're adding.

The scope of a *property*:*value*, or how far-reaching the style is, can be almost anything from one tiny letter or character on one page in the site to every character on every page in the site. There are many ways to control the scope of a CSS style, as you'll learn throughout this course. But suffice it to say, CSS gives you *very* precise control over styling, which is one of the reasons for its great success as a language.

In CSS3, you have about 250 properties to choose from and countless values (including over 16 million colors). But don't worry about all of that right now, and don't let it intimidate you. You'll learn about the important property:value pairs as we go through this course, and I'll show you easy ways to find whatever you need to know, whenever you need to know it. For now, it's important to just stay focused on the important fundamentals, including the terminology and syntax we're discussing right now.

Let's head over to Chapter 5 and wrap up what you've learned.