

Intro to CSS

STYLING HTML

HTML might credibly be called "the language of the web". But, if we're honest, it's pretty boring stuff. It can't do much of anything without JavaScript. So, is it all good looks and no brains? Well — no, it's awfully *plain* as it is. In a minute, we're going to ask you to go to a website, but first, here's a sample of that site without any CSS styling:

The screenshot shows a web browser window with a blank white page. The browser's address bar displays "file:///Users/halhelms/Downloads/index%20(2).html". The browser interface includes standard controls like back, forward, and search, along with a toolbar with various icons.

CSS Zen Garden

The Beauty of CSS Design

A demonstration of what can be accomplished through CSS-based design. Select any style sheet from the list to load it into this page.

Download the example [html file](#) and [css file](#)

The Road to Enlightenment

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, broken CSS support, and abandoned browsers.

We must clear the mind of the past. Web enlightenment has been achieved thanks to the tireless efforts of folk like the [W3C](#), [WaSP](#), and the major browser creators.

The CSS Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with clarity. Learn to use the time-honored techniques in new and invigorating fashion. Become one with the web.

So What is This About?

There is a continuing need to show the power of CSS. The Zen Garden aims to excite, inspire, and encourage participation. To begin, view some of the existing designs in the list. Clicking on any one will load the style sheet into this very page. The HTML remains the same, the only thing that has changed is the external CSS file. Yes, really.

CSS allows complete and total control over the style of a hypertext document. The only way this can be illustrated in a way that gets people excited is by demonstrating what it can truly be, once the reins are placed in the hands of those able to create beauty from structure. Designers and coders alike have contributed to the beauty of the web; we can always push it further.

Participation

Strong visual design has always been our focus. You are modifying this page, so strong CSS skills are necessary too, but the example files are commented well enough that even CSS novices can use them as starting points. Please see the [CSS Resource Guide](#) for advanced tutorials and tips on working with CSS.

You may modify the style sheet in any way you wish, but not the HTML. This may seem daunting at first if you've never worked this way before, but follow the listed links to learn more, and use the sample files as a guide.

Download the sample [HTML](#) and [CSS](#) to work on a copy locally. Once you have completed your masterpiece (and please, don't submit half-finished work) upload your CSS file to a web server under your control. [Send us a link](#) to an archive of that file and all associated assets, and if we choose to use it we will download it and place it on our server.

Benefits

Why participate? For recognition, inspiration, and a resource we can all refer to showing people how amazing CSS really can be. This site serves as equal parts inspiration for those working on the web today, learning tool for those who will be tomorrow, and gallery of future techniques we can all look forward to.

Requirements

Where possible, we would like to see mostly CSS 1 & 2 usage. CSS 3 & 4 should be limited to widely-supported elements only, or strong fallbacks should be provided. The CSS Zen Garden is about functional, practical CSS and not the latest bleeding-edge tricks viewable by 2% of the browsing public. The only real requirement we have is that your CSS validates.

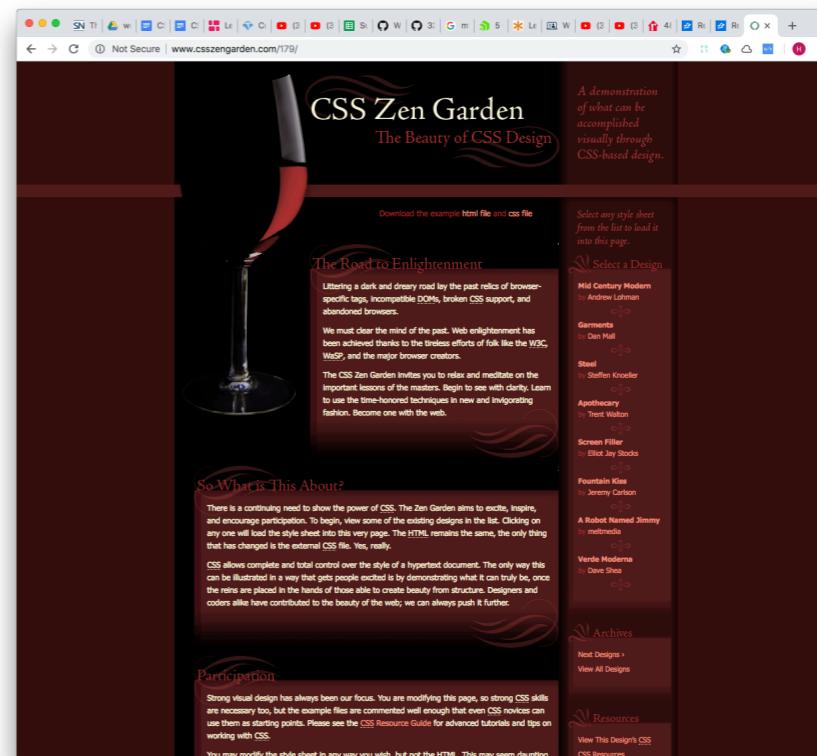
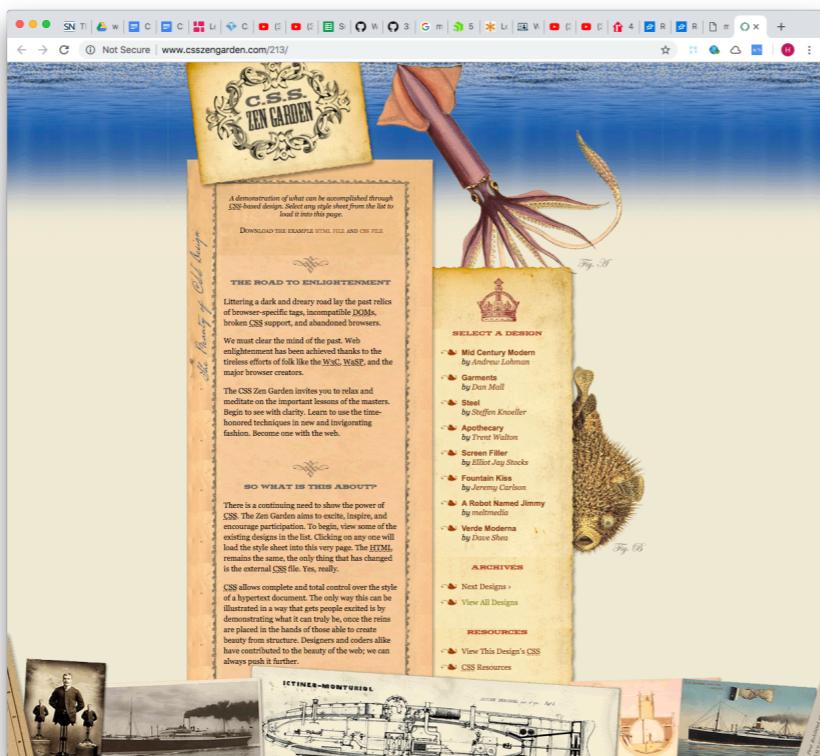
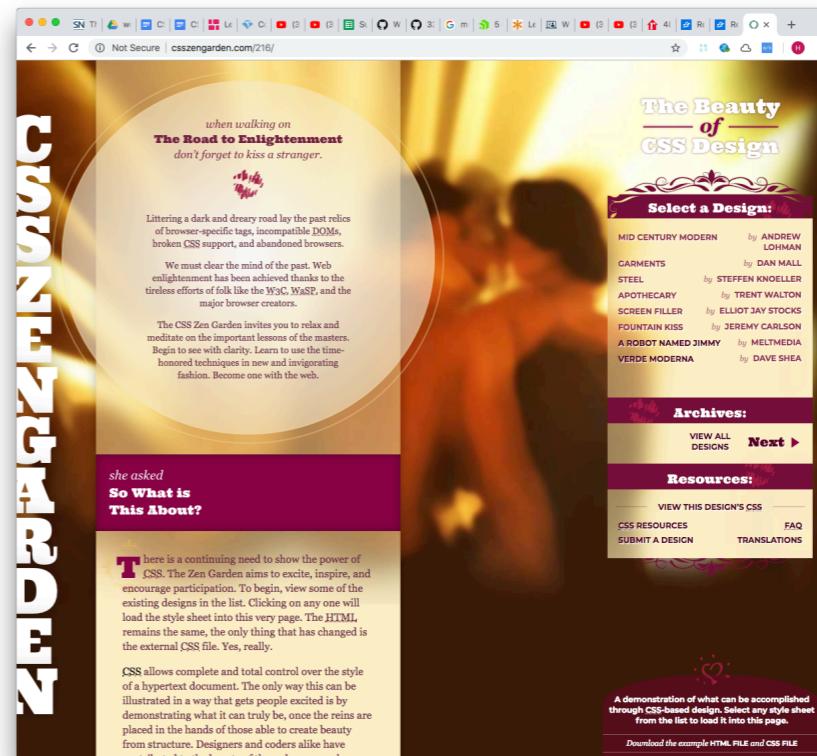
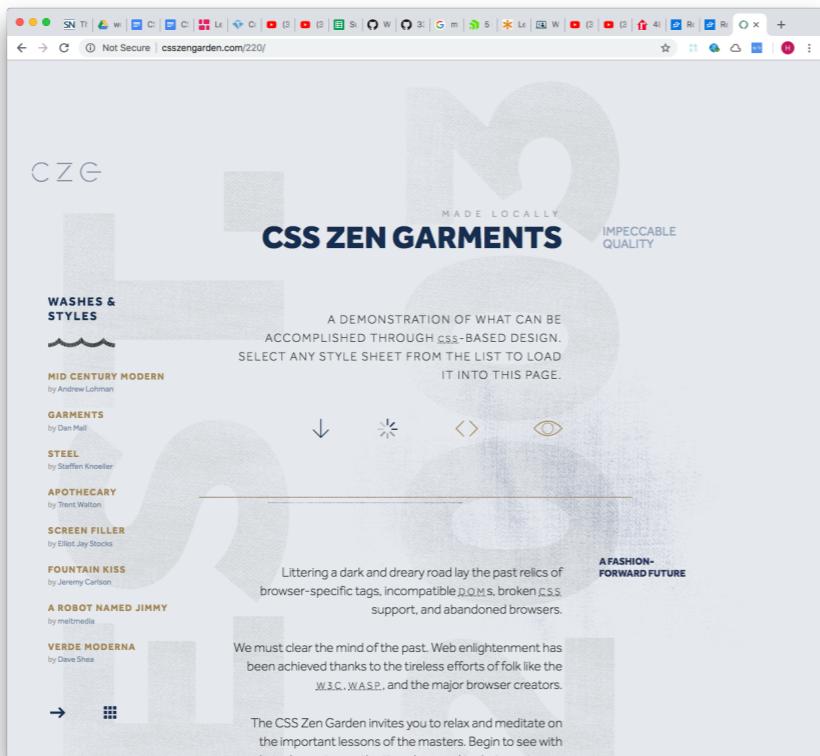
Luckily, designing this way shows how well various browsers have implemented CSS by now. When sticking to the guidelines you should see fairly consistent results across most modern browsers. Due to the sheer number of user agents on the web these days — especially when you factor in mobile — pixel-perfect layouts may not be possible across every platform. That's okay, but do test in as many as you can. Your design should work in at least IE9+ and the latest Chrome, Firefox, iOS and Android browsers (run by over 90% of the population).

We ask that you submit original artwork. Please respect copyright laws. Please keep objectionable material to a minimum, and try to incorporate unique and interesting visual themes to your work. We're well past the point of needing another garden-related design.

This is a learning exercise as well as a demonstration. You retain full copyright on your graphics (with limited exceptions, see [submission guidelines](#)), but we ask you release your CSS under a Creative Commons license identical to the [one on this site](#) so that others may learn from your work.

It provides structure (HTML is often compared to a human skeleton) but beauty? But here are snapshots of that very HTML — but with CSS added to it.

These four web pages all use exactly the same HTML. The only difference is the CSS applied to them.



These pages come from the site csszengarden.com. We urge you to go to the site, where you'll see over 200 different versions of that same page — all using the same markup, but different styling. That's the kind of impact CSS can have.

CSS RULESET

CSS styling is essentially simple: you *select* an element and then *style* the element by providing rules that guide its appearance. These are bundled together in a CSS *ruleset*. The following are various rulesets to give you a sense of how CSS is used. We'll look at specific selectors in subsequent sections of this Guide.

```
p {  
    color: red;  
}  
  
div#main {  
    font-family: "Arial";  
}  
  
li.active {  
    border: 1px solid grey;  
}  
  
div.special>p {  
    background-color: blue;  
    color: white;  
}
```

Selects all p elements

Sets text color to red

Selects the div element with an id of "main"

Sets font to "Arial"

Selects all li elements with a class of "active"

Sets a 1 pixel wide solid grey border around the element

Selects all p tags that are children of div tags with a class of "special"

Makes the background color blue and the text color white

REFERENCE HTML

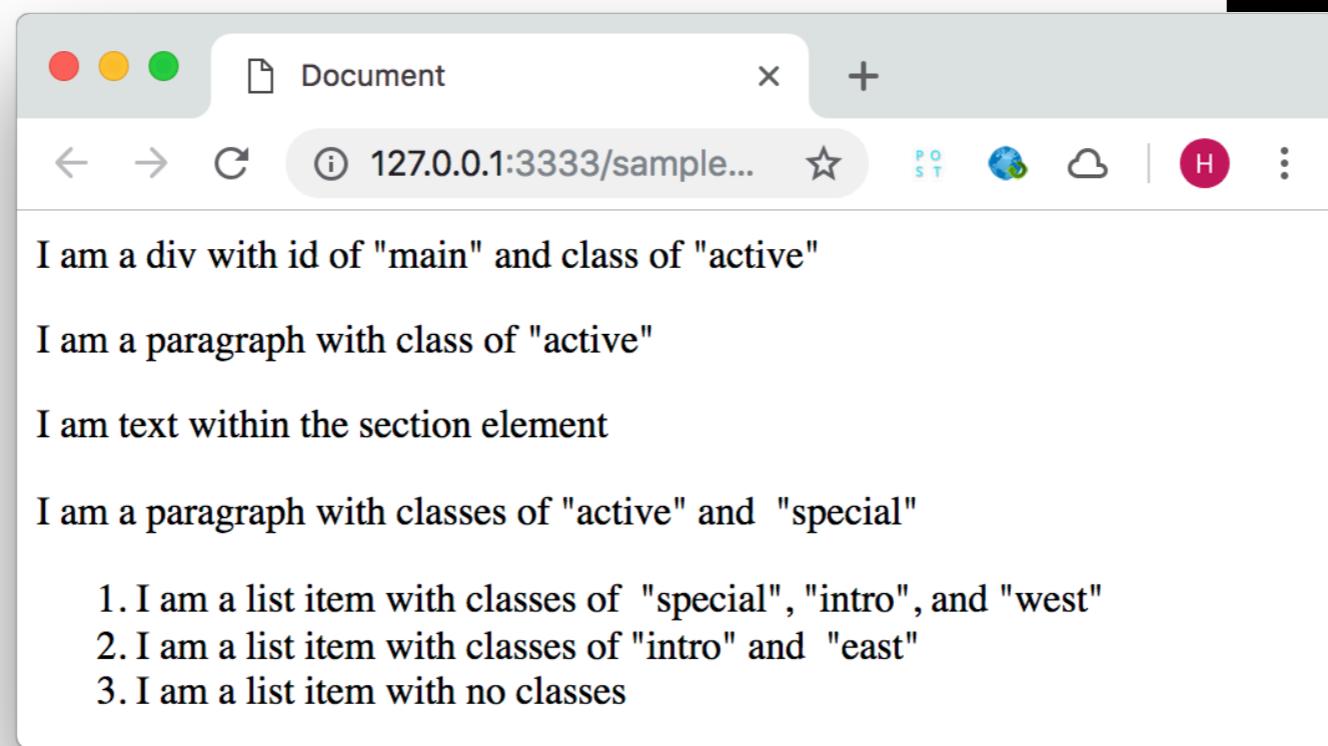
The following HTML will be used when working through various selectors.

Directories form a hierarchy of parents, children, grandchildren, etc. The "root" directory is the ultimate parent directory and, itself, has no parent.

```
<body>
  <div id="main" class="active">I am a div with id of "main" and class of "active"</div>
  <p class="active">I am a paragraph with class of "active"</p>
  <section>
    I am text within the section element
    <p class="active special">I am a paragraph with classes of "active" and "special"</p>
    <ol>
      <li class="special intro west">I am a list item with classes of "special", "intro", and "west"</li>
      <li class="intro east">I am a list item with classes of "intro" and "east"</li>
      <li>I am a list item with no classes</li>
    </ol>
  </section>
</body>
```

Although CSS is, as we said, simple, there are an enormous number of combinations of things you can do with it. Rather than trying to learn it all at once, we encourage you to learn things a little at a time, gradually acquiring mastery. In that spirit, our exploration of CSS will *not* attempt to be encyclopedic. We urge you to approach learning CSS in a spirit of *exploration*. When you see some effect you're baffled by, see how the designers performed their CSS magic.

Above HTML rendered
by the browser



TYPE SELECTORS

The following pages show various type selectors. All are styled to have a 2 pixel solid red outline in order to make the selection clear.

I am a div with id of "main" and class of "active"

I am a paragraph with class of "active"

I am text within the section element

I am a paragraph with classes of "active" and "special"

1. I am a list item with classes of "special", "intro", and "west"
2. I am a list item with classes of "intro" and "east"
3. I am a list item with no classes

```
p {  
    outline: 2px solid red;  
}
```

All <p> elements

I am a div with id of "main" and class of "active"

I am a paragraph with class of "active"

I am text within the section element

I am a paragraph with classes of "active" and "special"

1. I am a list item with classes of "special", "intro", and "west"
2. I am a list item with classes of "intro" and "east"
3. I am a list item with no classes

```
li {  
    outline: 2px solid red;  
}
```

All elements

CLASS SELECTORS

The following shows various class selectors.

A screenshot of a web browser window titled "Document". The address bar shows "127.0.0.1:3333/sample...". The page content includes several text elements:

- I am a div with id of "main" and class of "active"
- I am a paragraph with class of "active"
- I am text within the section element
- I am a paragraph with classes of "active" and "special" (highlighted with a red border)
- 1. I am a list item with classes of "special", "intro", and "west" (highlighted with a red border)
- 2. I am a list item with classes of "intro" and "east"
- 3. I am a list item with no classes

```
.special {  
    outline: 2px solid red;  
}
```

All elements of any type
with a class of "special"

A screenshot of a web browser window titled "Document". The address bar shows "127.0.0.1:3333/sample...". The page content includes several text elements:

- I am a div with id of "main" and class of "active" (highlighted with a red border)
- I am a paragraph with class of "active" (highlighted with a red border)
- I am text within the section element
- I am a paragraph with classes of "active" and "special" (highlighted with a red border)
- 1. I am a list item with classes of "special", "intro", and "west" (highlighted with a red border)
- 2. I am a list item with classes of "intro" and "east"
- 3. I am a list item with no classes

```
.active {  
    outline: 2px solid red;  
}
```

All elements of any type
with a class of "active"

ID SELECTORS

The following shows various id selectors.

While many elements can have classes with the same value/s, all id values on the same page must be unique.

You could have another element with an id, but its value could not be "main".

I am a div with id of "main" and class of "active"

I am a paragraph with class of "active"

I am text within the section element

I am a paragraph with classes of "active" and "special"

1. I am a list item with classes of "special", "intro", and "west"
2. I am a list item with classes of "intro" and "east"
3. I am a list item with no classes

```
#main {
    outline: 2px solid red;
}
```

Element with an id of "main"

UNIVERSAL SELECTOR

I am a div with id of "main" and class of "active"

I am a paragraph with class of "active"

I am text within the section element

I am a paragraph with classes of "active" and "special"

1. I am a list item with classes of "special", "intro", and "west"
2. I am a list item with classes of "intro" and "east"
3. I am a list item with no classes

```
*{
    outline: 2px solid red;
}
```

All elements

DESCENDANT SELECTORS

There are multiple ways of getting to the same elements.

As an example, the list items (``) can be selected either as a simple type selector or a descendant selector.

As a general strategy, the more specific selector (descendants in this case) is a better plan.

I am a div with id of "main" and class of "active"

I am a paragraph with class of "active"

I am text within the section element

I am a paragraph with classes of "active" and "special"

1. I am a list item with classes of "special", "intro", and "west"

2. I am a list item with classes of "intro" and "east"

3. I am a list item with no classes

```
section>ol>li{  
    outline: 2px solid red;  
}
```

`` elements that are children of `` that, itself, is a child of `<section>`

PSEUDO-CLASS SELECTORS

One of the more interesting selector strategies is the use of *pseudo-classes*. Unlike an explicit class selector, pseudo-classes reflect the state of an element. Some example pseudo-selectors are...

- `:active` — an element activated by the user (such as mouse click on that selector)
- `:first-child` — the first element in a group of sibling elements
- `:focus` — an element (typically a form field) that currently has the focus
- `:hover` — an element the mouse hovers over

The use of pseudo-classes is difficult to represent in Guide format. Check the [Intro to CSS Selectors](#) video to see the use of pseudo-classes.