

CSS Form

Web application with HTML and CSS



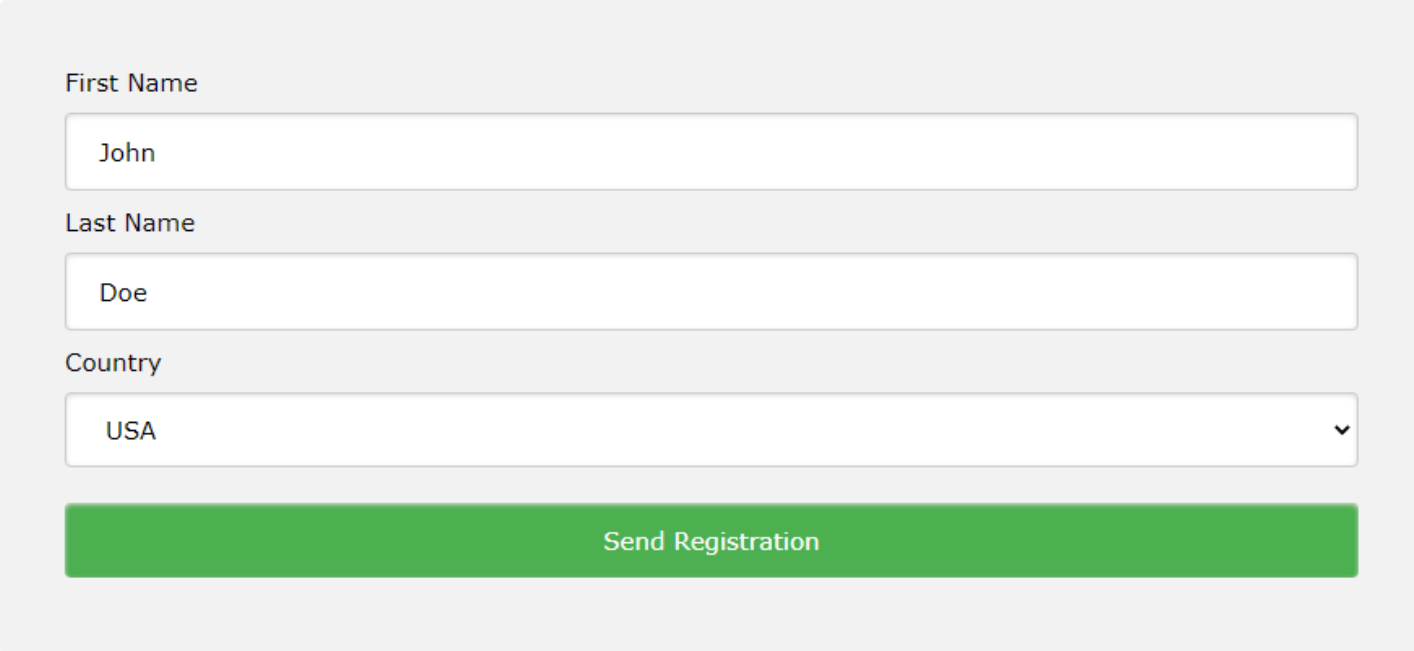
Lesson Objectives

- CSS Form
- Debug CSS
- Minimize CSS
- Use CSS library

Section 1

CSS FORM

The look of an HTML form can be greatly improved with CSS:



First Name

Last Name

Country

Send Registration

- Almost form elements are input, with difference types
- User focused in elements to input data
- Make sure all form elements can be click or touch easily
- Make all form elements are **clear** and **clean**

- Use CSS attribute selector

- ✓ `input[type="text"]`
- ✓ `input[type="password"]`
- ✓ `input[type="number"]`
- ✓ `input[type="email"]`
- ✓ `input[type="tel"]`
- ✓ `input[type="checkbox"]`
- ✓ `input[type="radio"]`

➤ Styling Input Fields:

- Use the ***width*** property to determine the width of the input field

```
input {  
  width:100%;  
}
```

- Use the ***padding*** property to add space inside the text field

```
input[type=text] {  
  padding: 10px;  
}
```

➤ Styling Input Fields:

- Use the ***border*** property to change the border size and color, and use the ***border-radius*** property to add rounded corners

```
.rounded-input {  
  padding:10px;  
  border-radius:10px;  
}
```



- Use the ***background-color*** property to add a background color to the input, and the ***color*** property to change the text color

```
input[type=text] {  
  background-color: #3CBC8D;  
  color: white;  
}
```



➤ Styling Input Fields:

- By default, some browsers will add a blue outline around the input when it gets focus (clicked on).
- You can remove this behavior by adding `outline: none;` to the input.
- Use the `:focus` selector to do something with the input field when it gets focus:

Example

```
input[type=text]:focus {  
  background-color: lightblue;  
}
```

Example

```
input[type=text]:focus {  
  border: 3px solid #555;  
}
```

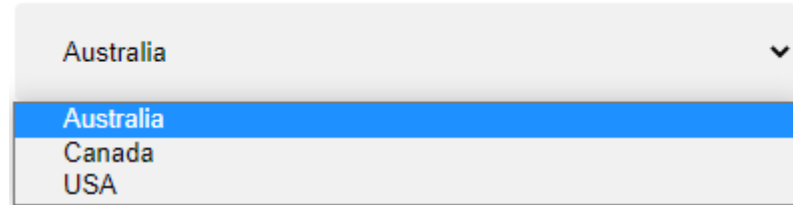
Styling forms

```
textarea {  
  width: 100%;  
  height: 150px;  
  padding: 12px 20px;  
  box-sizing: border-box;  
  border: 2px solid #ccc;  
  border-radius: 4px;  
  background-color: #f8f8f8;  
  resize: none;  
}
```

Some text...

Styling forms

```
select {  
  width: 100%;  
  padding: 16px 20px;  
  border: none;  
  border-radius: 4px;  
  background-color: #f1f1f1;  
}
```



Styling forms

```
input[type=button], input[type=submit],  
input[type=reset] {  
    background-color: #4CAF50;  
    border: none;  
    color: white;  
    padding: 16px 32px;  
    text-decoration: none;  
    margin: 4px 2px;  
    cursor: pointer;  
}
```

/ Tip: use **width: 100%** for full-width buttons */*



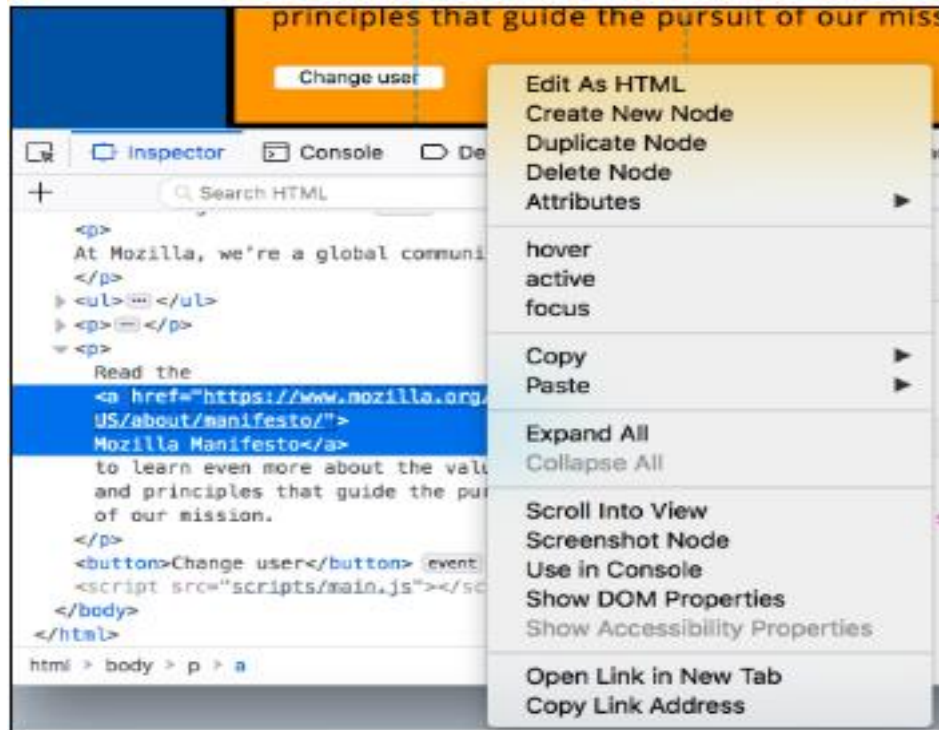
Section 2

DEBUG CSS

- Every modern web browser includes a powerful suite of **developer tools**. These tools do a range of things, from inspecting currently-loaded HTML, CSS and JavaScript to showing which assets the page has requested and how long they took to load
- The **devtools** live inside your browser in a subwindow that looks roughly like this, depending on what browser you are using

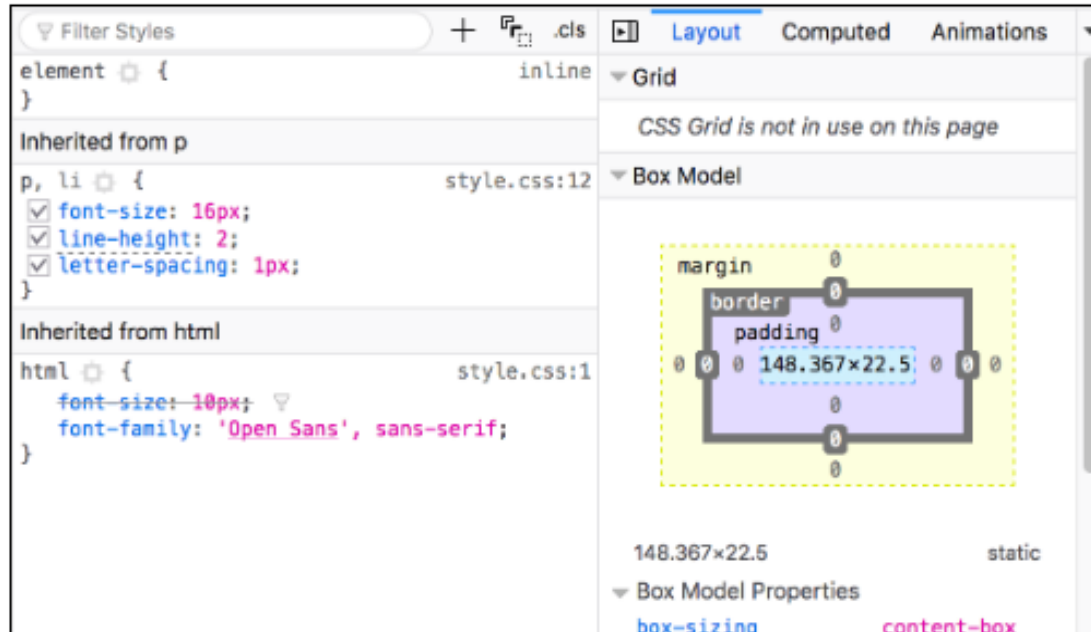
Inspecting the applied CSS

➤ Exploring the DOM inspector



Inspecting the applied CSS

- **Exploring the CSS editor:** By default, the CSS editor displays the CSS rules applied to the currently selected element



➤ Exploring the CSS editor

- The rules applied to the current element are shown in order of most-to-least-specific.
- Click the checkboxes next to each declaration to see what would happen if you removed the declaration.
- Click the little arrow next to each shorthand property to show the property's longhand equivalents.
- Click a property name or value to bring up a text box, where you can key in a new value to get a live preview of a style change.

➤ Exploring the CSS editor

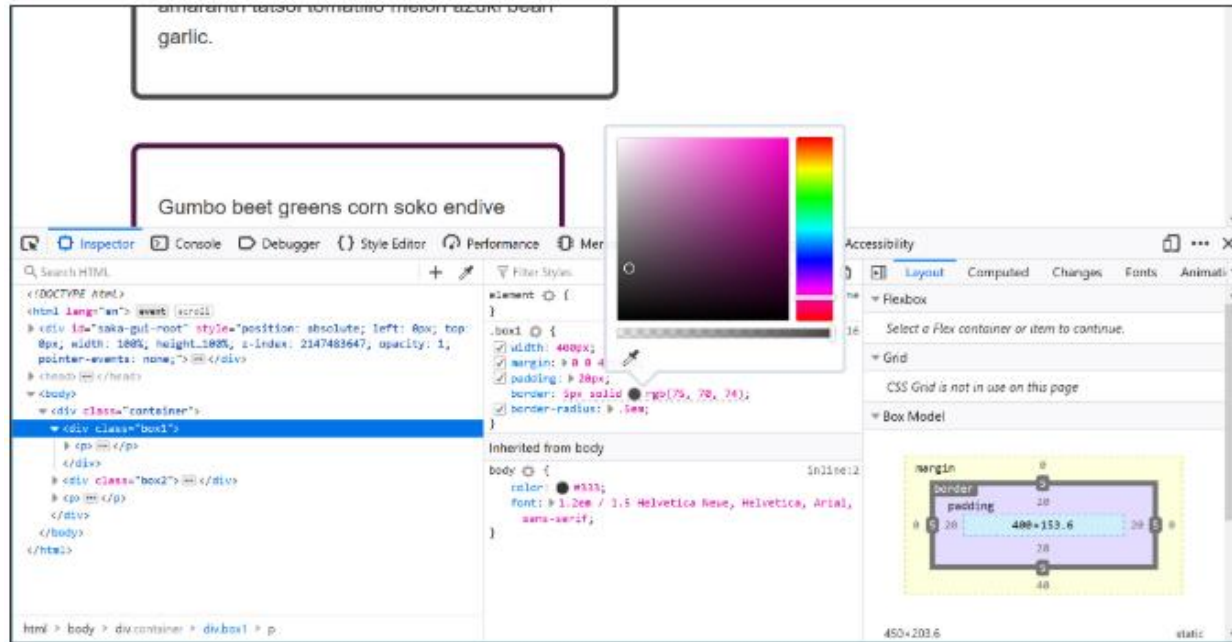
- Next to each rule is the file name and line number the rule is defined in. Clicking that rule causes the dev tools to jump to show it in its own view, where it can generally be edited and saved.
- You can also click the closing curly brace of any rule to bring up a text box on a new line, where you can write a completely new declaration for your page

➤ Exploring the CSS editor – Clickable tab

- **Computed:** This shows the computed styles for the currently selected element (the final, normalized values that the browser applies).
- **Layout:** In Firefox, this area includes two sections:
- **Box Model:** Represents visually the current element's box model, so you can see at a glance what padding, border and margin is applied to it, and how big its content is.
- **Grid:** If the page you are inspecting uses CSS Grid, this section allows you to view the grid details.
- **Fonts:** In Firefox, the Fonts tab shows the fonts applied to the current element.

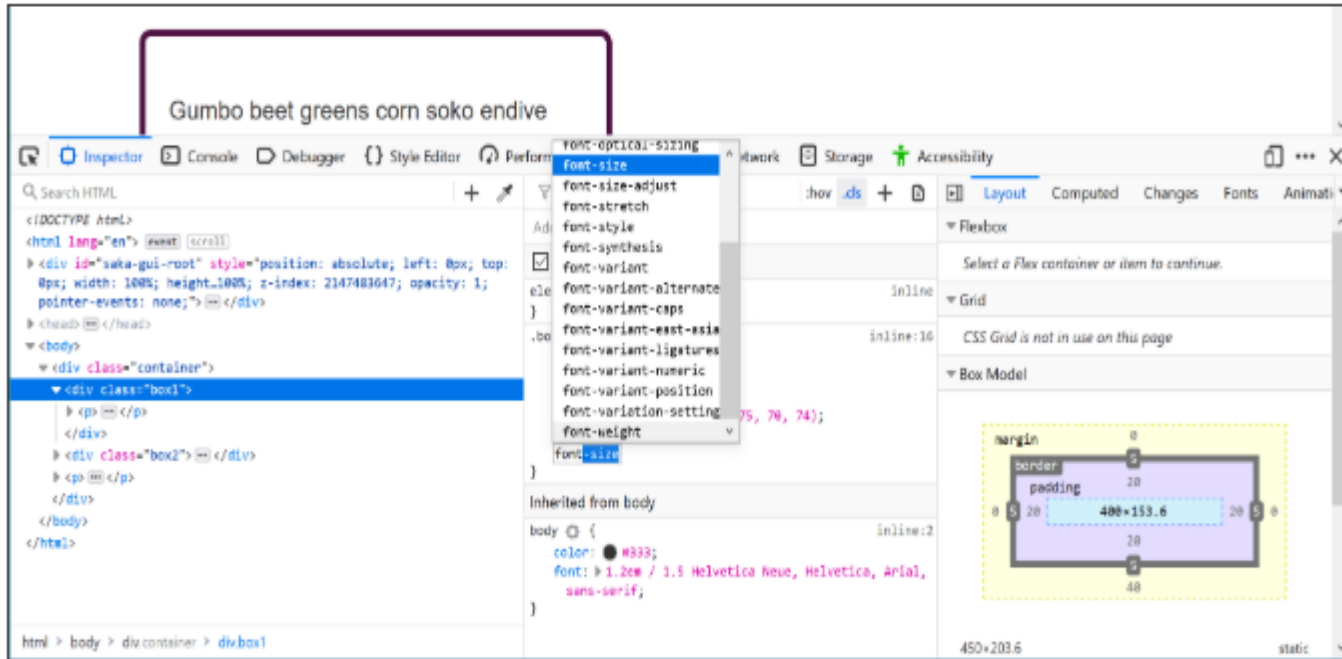
Editing value

- In addition to turning properties on and off, you can **edit** their values, DevTools can save you a lot of time editing a stylesheet and reloading the page.



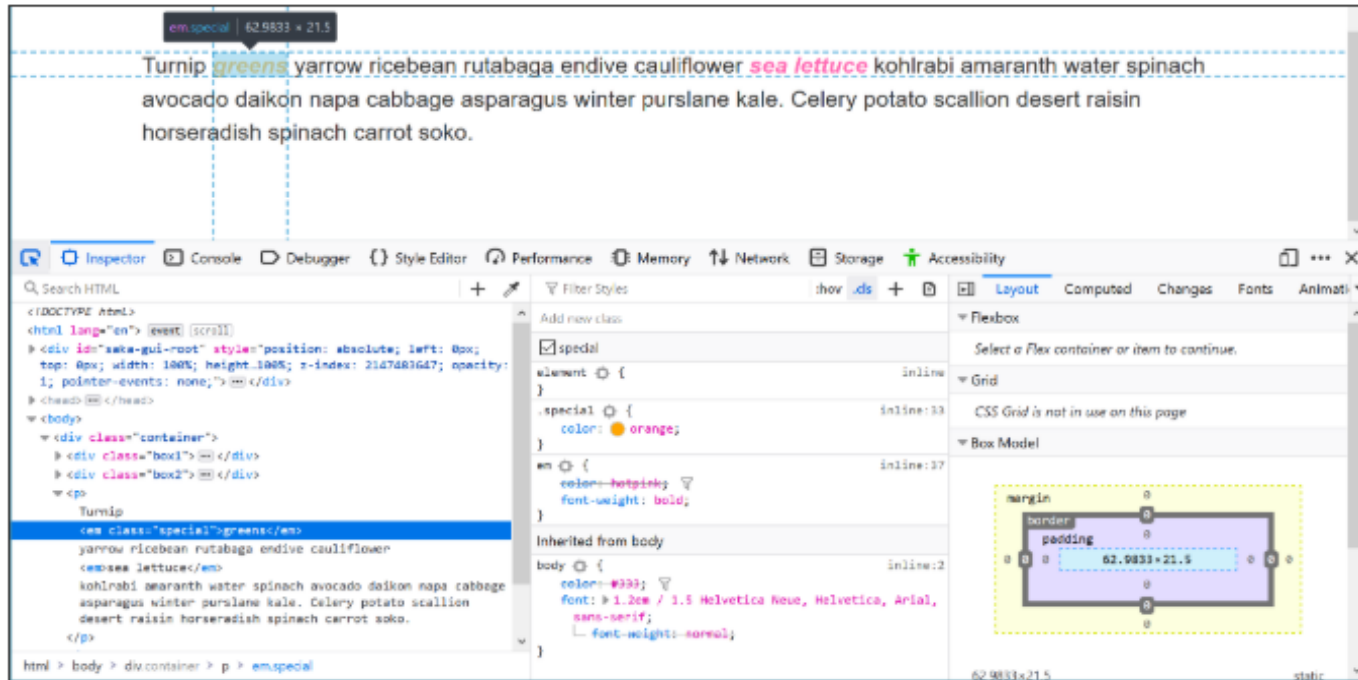
Adding a new property

- You can add properties using the DevTools. You can try this out in DevTools before adding it to your CSS file.



Solving specificity problem

- We can solve specificity problem with DevTool by inspecting element to see which style is applied to element.



➤ **Some tips for debugging problems in CSS**

- Take a step back from the problem
- Do you have valid HTML and CSS?
- Is the property and value supported by the browser you are testing in?
- Is something else overriding your CSS?
- Make a reduced test case of the problem

Section 3

CSS MINIFICATION

- CSS code comes from many sources
- More code => More complex
- Longer code => lower load

- Minification helps to cut out unnecessary portions of code and reduce its file size.
- CSS minification allows to strip out these extras and apply a number of optimizations so that shipping just what the computer needs to execute on the target device

- Reduce file size => Faster load
- Reduce http/https request
- Hide CSS code

Minify vs. compress

- **Minification** alters the content of code. It reduces code file size by stripping out unwanted spaces, characters, and formatting, resulting in fewer characters in the code. It may further optimize the code by safely renaming variables to use even fewer characters.
- **Compression** does not necessarily alter the content of code — well, unless we consider binary files like images, which we are not covering in this exploration. It reduces file size by compacting the file before serving it to the browser when it is requested

- Code minification and compression are often used interchangeably, maybe because they both address performance optimizations that lead to size reductions.
- These two techniques are not mutually exclusive, so they can be used together to deliver optimized code to the user.

CSS minification

```
html,
body {
  height: 100%;
}
body {
  padding: 0;
  margin: 0;
}
body .pull-right {
  float: right !important;
}
body .pull-left {
  float: left !important;
}
```

Summary

Characters (without line endings): 147
Words: 21
Lines: 14
Document length: 173
0 selected characters (0 bytes) in 0 ranges

OK

```
html,body{height:100%}body{padding:0;margin:0}bo
dy .pull-right{float:right!important}body .pull-
left{float:left!important}
```

Select Language: ☐ JS ☒ CSS

Minify

Original script: 173b, minified script: 122b. Gain: 51b.

Standalone online tools

- [Minify](#)
- [CSS Minifier](#)
- [Minify Code](#)

Practice

CSS MINIFICATION

Section 4

USE LIBRARIES

- Speeds up development
- Enables cross-browser functionality
- Enforces good web design habits
- Gives clean and symmetrical layouts
- Make styling workflow productive, clean, and maintainable

Use libraries

- Use cdn (online)
- Download and use (offline)

- **Web fonts** are a CSS feature that allows you to specify font files to be downloaded along with your website as it is accessed.
- **@font-face:** First of all, you have a @font-face block at the start of the CSS, which specifies the font file(s) to download:

```
1  @font-face {  
2      font-family: "myFont";  
3      src: url("myFont.woff");  
4  }
```

- **@font-face:** To use the font for an HTML element, refer to the name of the font (myFirstFont) through the font-family property

```
1 | html {  
2 |     font-family: "myFont", "Bitstream Vera Serif", serif;  
3 | }
```

➤ Steps to use a web font

1. Finding fonts
2. Generating the required code
3. Implementing the code in your demo

- Icons are scalable without loss in definition
- There's no need to worry about Retina displays
- It's easy to apply CSS properties without editing the icon itself (color, gradient, shadows, etc.)
- Can use the same icon in different sizes and colors to save time and space
- Better page speed performance (i.e. fewer http requests)
- Icon fonts load faster than background or inline SVG's

- Font Awesome
 - ✓ <https://fontawesome.com/>
- Bootstrap Icons
 - ✓ <https://icons.getbootstrap.com/>
- Icons Reference
 - ✓ https://www.w3schools.com/icons/icons_reference.asp

Icons Reference

Q heart

Show icons from:

- ☒ Font Awesome 5
- ☒ Font Awesome 4
- ☒ Bootstrap
- ☒ Google

Font Awesome 5:

	fas fa-grin-hearts		Try it
	far fa-grin-hearts		Try it
	fas fa-hand-holding-heart		Try it
	fas fa-heart		Try it
	far fa-heart		Try it
	fas fa-heart-broken		Try it
	fas fa-heartbeat		Try it
	fas fa-kiss-wink-heart		Try it
	far fa-kiss-wink-heart		Try it

Lesson Summary



Thank you

