Lesson #6

Fonts, More CSS Selectors, CSS Specificity

Introduction to Web Development

Prof. Nayeon Kim
Afghan Pathways Program | Viterbi School of Engineering

USC Viterbi
School of Engineering

University of Southern California

Fonts

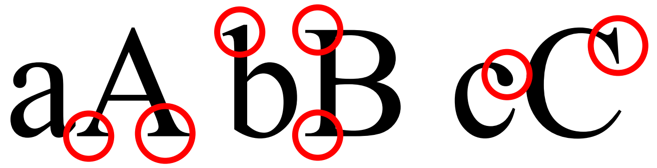
Typefaces (Font Families)

5 popular web typefaces:

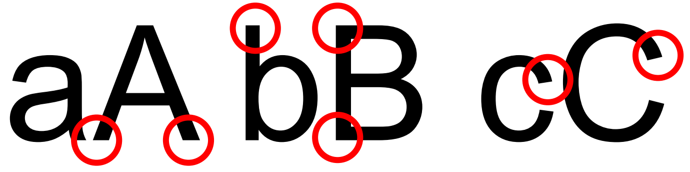
Serif	Fonts with small lines (serifs) attached at the end of strokes in letters.
Sans-Serif	Fonts without serifs (small lines).
Monospace	Fonts with letters & characters each occupying same amount of horizontal space.
Cursive	Fonts that emulate handwriting.
FANTASY	DECORATIVE FONTS.

Serif vs Sans-Serif Typefaces

Serif
(Times New Roman)



Sans-Serif
(Arial)



Serif vs Sans-Serif Typefaces

Google

2013 - 2015

Google

2015 - Present

Monospace Typeface

```
aA bB cC
```

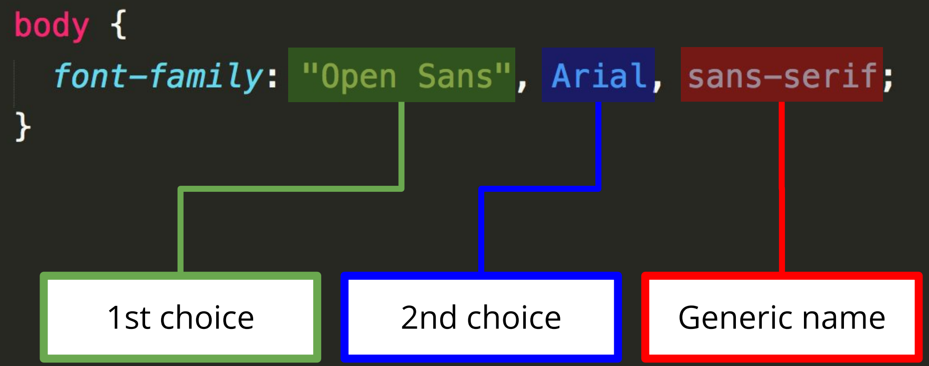
```
University of Southern California
```

CSS `font-family`

Specifies typefaces to be applied in prioritized order.

Always include generic typeface at the end.

Use quotations for font names with more than 1 word.



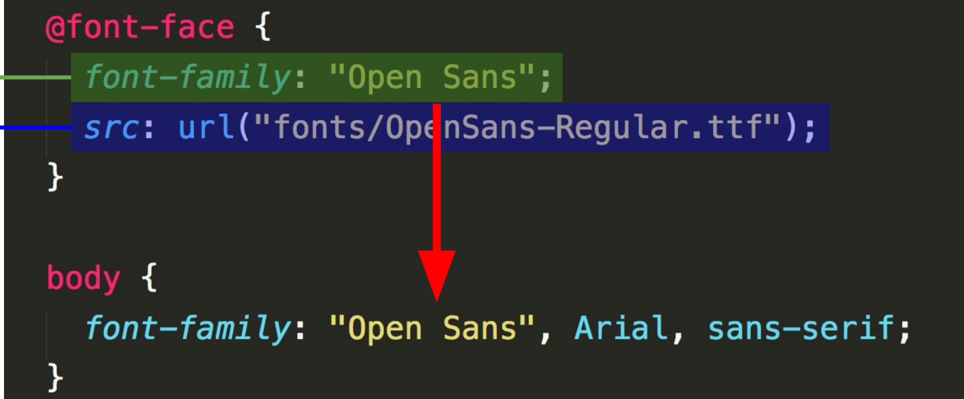
```
body {
  font-family: "Open Sans", Arial, sans-serif;
}
```

CSS `@font-face` rule

Loads custom fonts.

Required descriptors:

1. Font name,
2. Font location.



```
@font-face {
  font-family: "Open Sans";
  src: url("fonts/OpenSans-Regular.ttf");
}

body {
  font-family: "Open Sans", Arial, sans-serif;
}
```

CSS Units

CSS Units

So far we've been using **absolute units** like **px** to set a font size, width, padding, margin, etc.

However, in a world where screen sizes can be so different for everyone, we want to use units that are **adaptable** as much as possible.

Introducing… **rem** and **em.**

```css
h1 {
    font-size: 40px;
}
```

```css
h1 {
    font-size: 1rem;
}
```

rem

- rem stands for "root em".
- One rem is equal to the font size of the **root element** (usually <html>).
- The root element defaults to 16px in most browsers, so 1rem is equal to 16px.
- 2rem = 32px
- The size is **relative** so if the root element is bigger, the size will grow proportionally.

```css
h1 {
    font-size: 2rem; /* 32px */
}

h2 {
    font-size: 1.5rem; /* 24px */
}
```

em

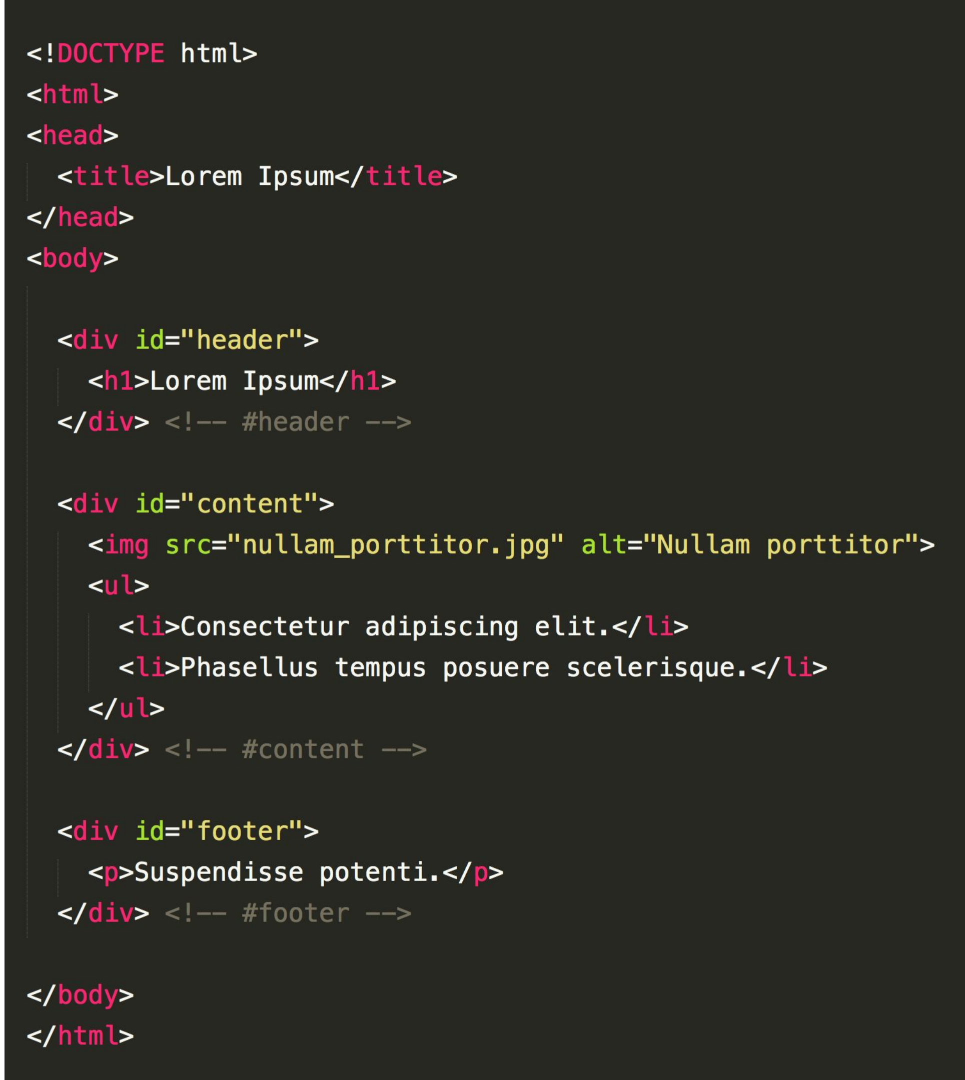
- Similar to `rem`, `em` is a relative unit of measurement.
- But unlike `rem`, `em` is relative to the font size of the **parent element** or the font size of the nearest parent with a defined font size.
- Useful when you need to scale an element to be consistent with the parent.
- Used a lot for margin and padding

```
#welcome {
    font-size: 20px;
}
#welcome p {
    font-size: 1em; /* 20px */
}


<div id="welcome">
    <p>🍵 Welcome!</p>
</div>
```

More CSS Selectors

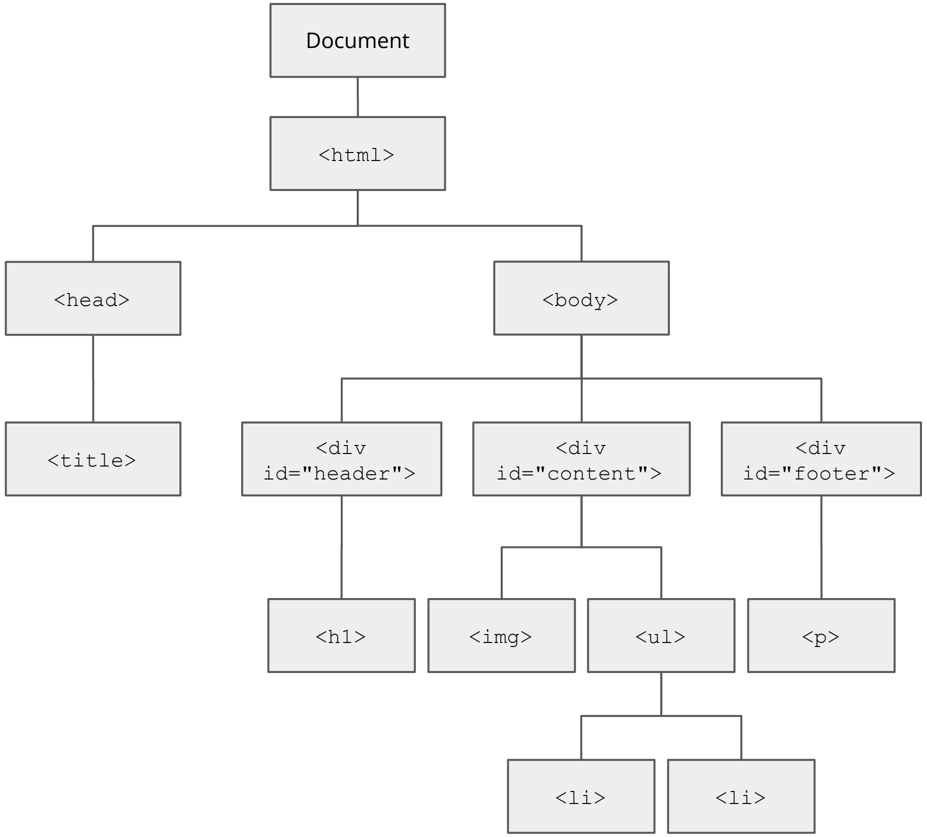
DOM

DOM – **D**ocument **O**bject **M**odel.
Cross-platform way of representing HTML objects in a document.



```html
<!DOCTYPE html>
<html>
<head>
  <title>Lorem Ipsum</title>
</head>
<body>

  <div id="header">
    <h1>Lorem Ipsum</h1>
  </div> <!-- #header -->

  <div id="content">
    <img src="nullam_porttitor.jpg" alt="Nullam porttitor">
    <ul>
      <li>Consectetur adipiscing elit.</li>
      <li>Phasellus tempus posuere scelerisque.</li>
    </ul>
  </div> <!-- #content -->

  <div id="footer">
    <p>Suspendisse potenti.</p>
  </div> <!-- #footer -->

</body>
</html>
```

DOM

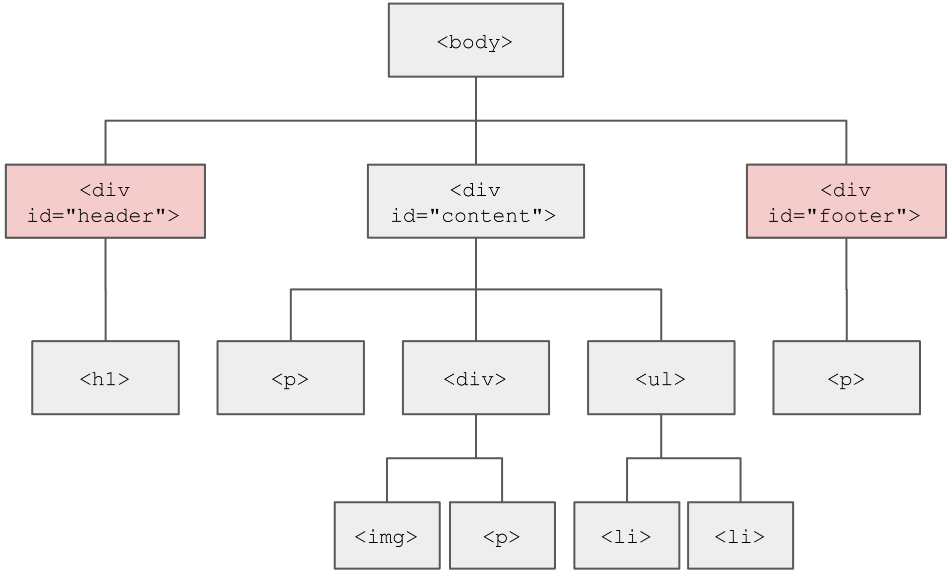


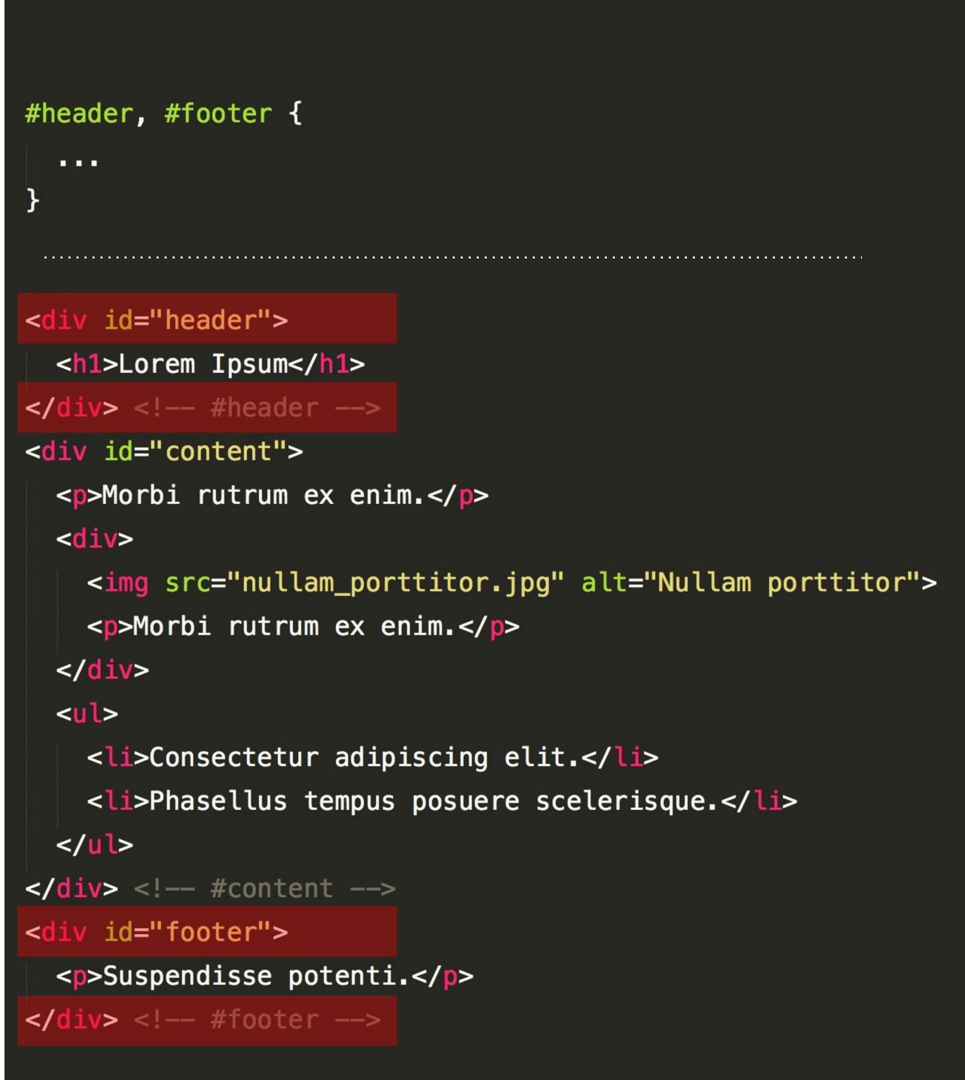
```html
<!DOCTYPE html>
<html>
<head>
  <title>Lorem Ipsum</title>
</head>
<body>

  <div id="header">
    <h1>Lorem Ipsum</h1>
  </div> <!-- #header -->

  <div id="content">
    <img src="nullam_porttitor.jpg" alt="Nullam porttitor">
    <ul>
      <li>Consectetur adipiscing elit.</li>
      <li>Phasellus tempus posuere scelerisque.</li>
    </ul>
  </div> <!-- #content -->

  <div id="footer">
    <p>Suspendisse potenti.</p>
  </div> <!-- #footer -->

</body>
</html>
```

Combinator CSS Selectors

`elt-1, elt-2`	`elt-1` and `elt-2`
`elt-1 elt-2`	`elt-2` that are descendants of `elt-1`
`elt-1 > elt-2`	`elt-2` with `elt-1` parent
`elt-1 + elt-2`	`elt-2` that is the immediate next sibling of `elt-1`

Combinator CSS Selectors

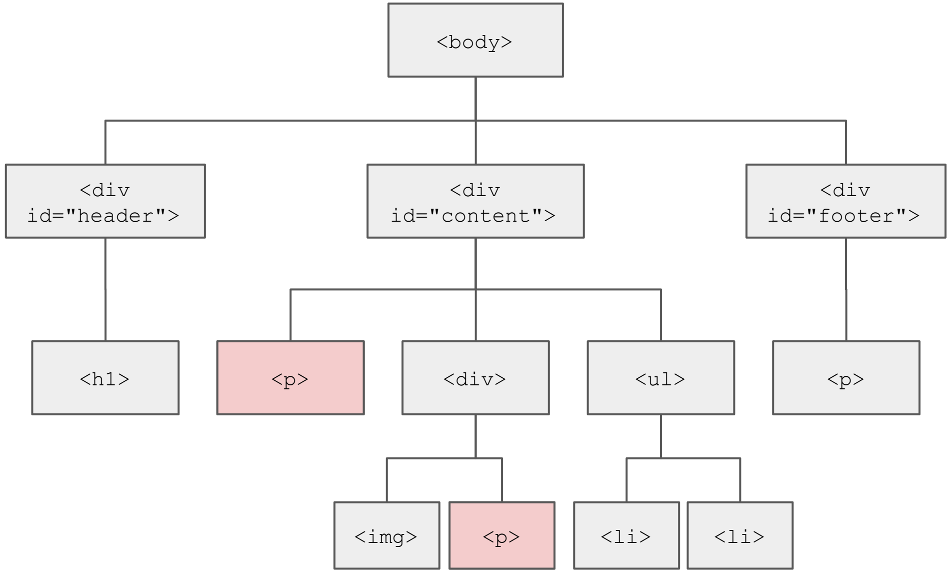




```
#header, #footer {
  ...
}
```

```
<div id="header">
  <h1>Lorem Ipsum</h1>
</div> <!-- #header -->
<div id="content">
  <p>Morbi rutrum ex enim.</p>
  <div>
    <img src="nullam_porttitor.jpg" alt="Nullam porttitor">
    <p>Morbi rutrum ex enim.</p>
  </div>
  <ul>
    <li>Consectetur adipiscing elit.</li>
    <li>Phasellus tempus posuere scelerisque.</li>
  </ul>
</div> <!-- #content -->
<div id="footer">
  <p>Suspendisse potenti.</p>
</div> <!-- #footer -->
```

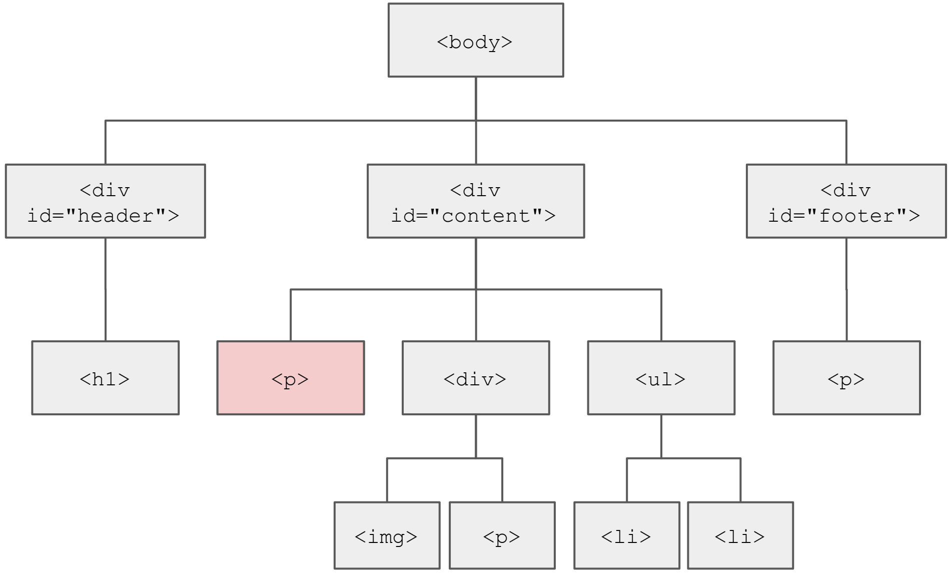
Combinator CSS Selectors

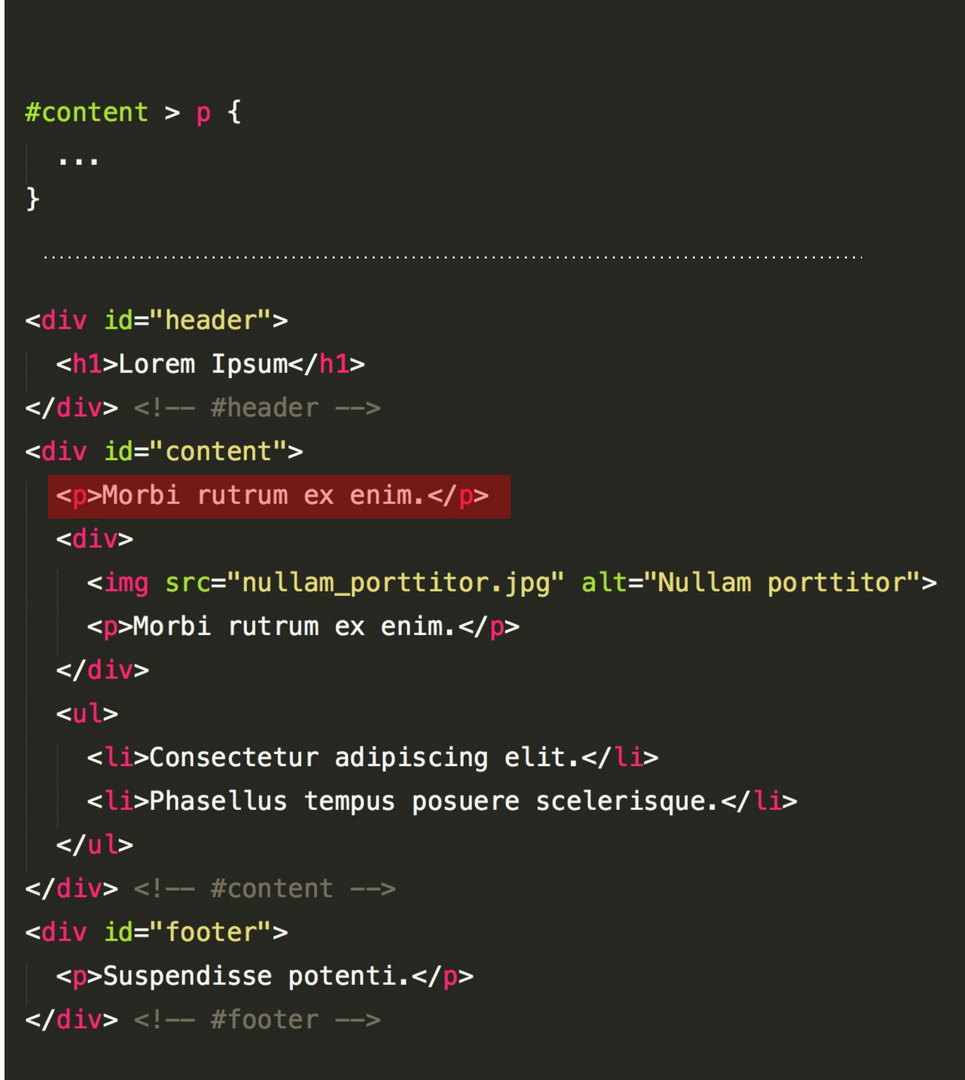


```
#content p {
  ...
}
```

```
<div id="header">
  <h1>Lorem Ipsum</h1>
</div> <!-- #header -->
<div id="content">
  <p>Morbi rutrum ex enim.</p>
  <div>
    <img src="nullam_porttitor.jpg" alt="Nullam porttitor">
    <p>Morbi rutrum ex enim.</p>
  </div>
  <ul>
    <li>Consectetur adipiscing elit.</li>
    <li>Phasellus tempus posuere scelerisque.</li>
  </ul>
</div> <!-- #content -->
<div id="footer">
  <p>Suspendisse potenti.</p>
</div> <!-- #footer -->
```

Combinator CSS Selectors

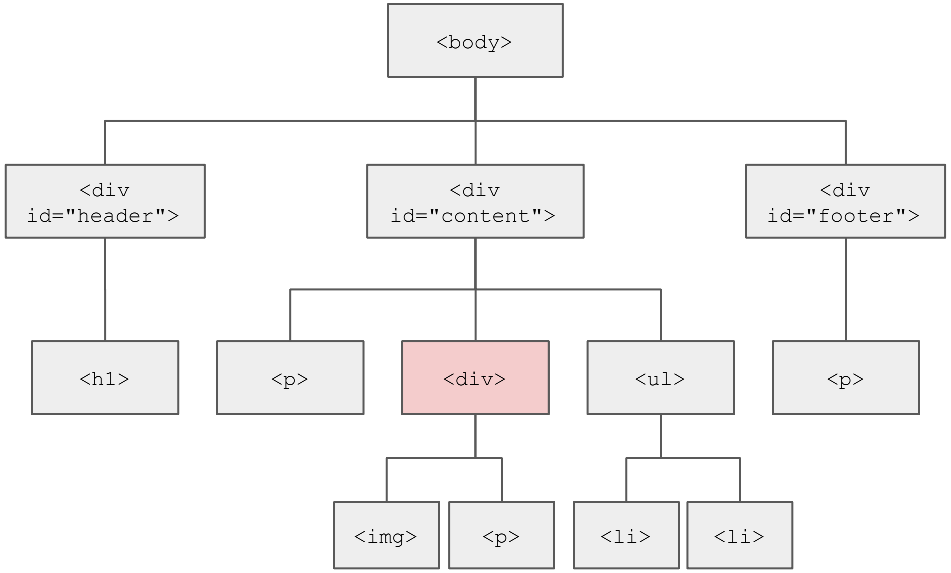


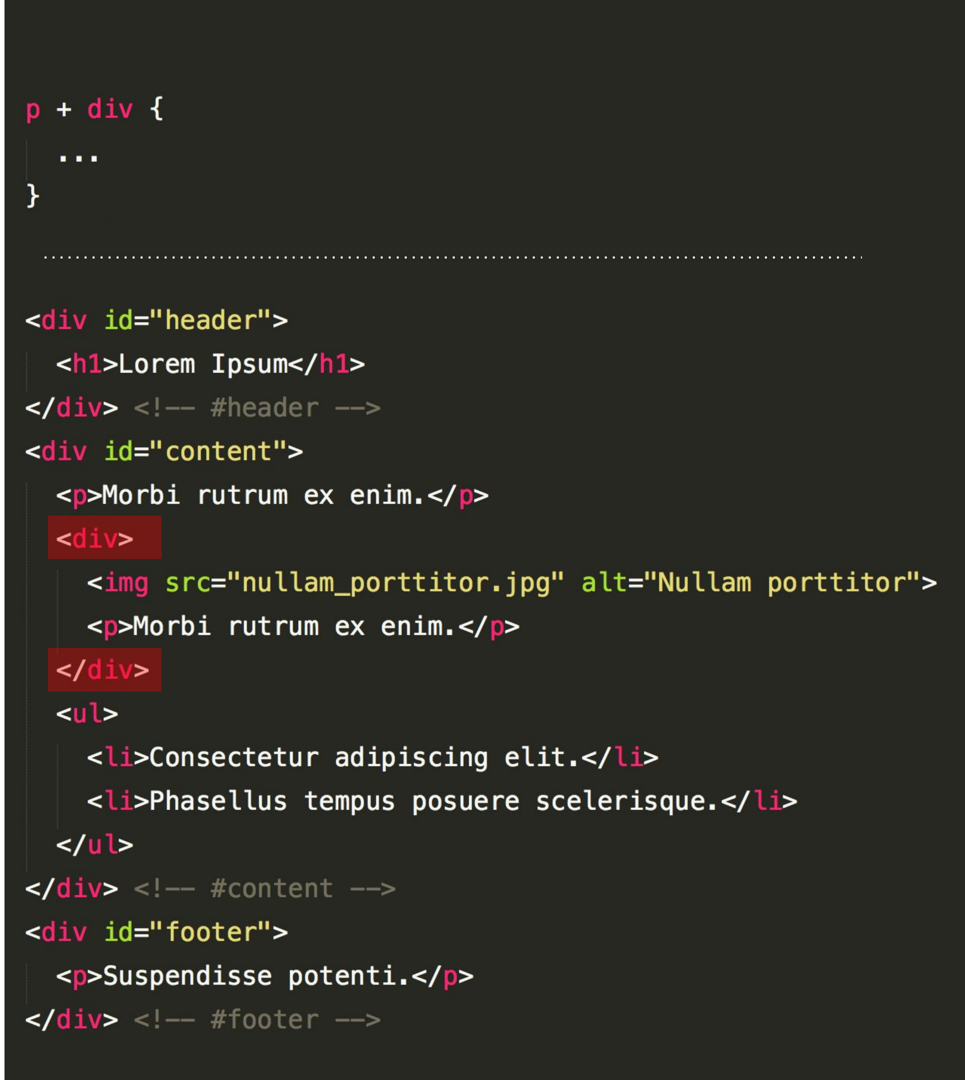


```
#content > p {
  ...
}
```

```
<div id="header">
  <h1>Lorem Ipsum</h1>
</div> <!-- #header -->
<div id="content">
  <p>Morbi rutrum ex enim.</p>
  <div>
    <img src="nullam_porttitor.jpg" alt="Nullam porttitor">
    <p>Morbi rutrum ex enim.</p>
  </div>
  <ul>
    <li>Consectetur adipiscing elit.</li>
    <li>Phasellus tempus posuere scelerisque.</li>
  </ul>
</div> <!-- #content -->
<div id="footer">
  <p>Suspendisse potenti.</p>
</div> <!-- #footer -->
```

Combinator CSS Selectors





```
p + div {
  ...
}
```

```
<div id="header">
  <h1>Lorem Ipsum</h1>
</div> <!-- #header -->
<div id="content">
  <p>Morbi rutrum ex enim.</p>
  <div>
    <img src="nullam_porttitor.jpg" alt="Nullam porttitor">
    <p>Morbi rutrum ex enim.</p>
  </div>
  <ul>
    <li>Consectetur adipiscing elit.</li>
    <li>Phasellus tempus posuere scelerisque.</li>
  </ul>
</div> <!-- #content -->
<div id="footer">
  <p>Suspendisse potenti.</p>
</div> <!-- #footer -->
```

CSS Specificity

CSS Specificity

- Rules that determine which CSS values are applied in case of conflict.
- Selectors in order of importance:
  - Inline styles
  - id,
  - class, attribute, pseudo-class
  - type, pseudo-element

Result:

Hello World!

```
p {
    background-color: green;
}
.example-class {
    background-color: orange;
}
```

```
<p id="example-id" class="example-class">
  Hello World!
</p>
```

CSS Specificity

- Rules that determine which CSS values are applied in case of conflict.
- Selectors in order of importance:
  - Inline styles
  - id,
  - class, attribute, pseudo-class
  - type, pseudo-element

Result:

Hello World!

```
p {
    background-color: green;
}
.example-class {
    background-color: blue;
}
#example-id {
    background-color: red;
}
```

```
<p id="example-id" class="example-class">
  Hello World!
</p>
```

CSS Specificity

- Rules that determine which CSS values are applied in case of conflict.
- Selectors in order of importance:
    - Inline styles
    - id,
    - class, attribute, pseudo-class
    - type, pseudo-element

Result:

Hello World!

```css
p {
    background-color: green;
}
.example-class {
    background-color: blue;
}
#example-id {
    background-color: red;
}
```

```html
<p style="background-color:yellow;">Hello
World!</p>
```

CSS Specificity

- Rules that determine which CSS values are applied in case of conflict.
- Selectors in order of importance:
  - Inline styles
  - id,
  - class, attribute, pseudo-class
  - type, pseudo-element

Result:

Hello World!

```css
p {
    background-color: green;
}
.purple{
    background-color: purple;
}
.pink {
    background-color: pink;
}
```

```html
<p id="example-id" class="pink purple">
  Hello World!
</p>
```

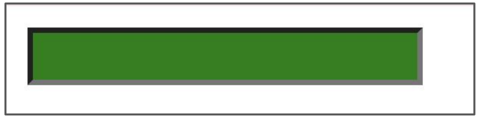
CSS Specificity

- Rules that determine which CSS values are applied in case of conflict.
- Selectors in order of importance:
    - Inline styles
    - id,
    - class, attribute, pseudo-class
    - type, pseudo-element

```css
input[type="email"] {
    background-color: green;
}
input {
    background-color: blue;
}
```

```html
<input type="email" class="input-class"
id="input-id" />
```

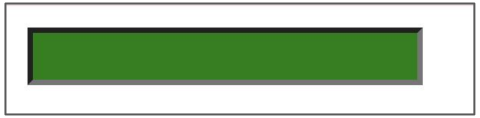
Result:



CSS Specificity

- Rules that determine which CSS values are applied in case of conflict.
- Selectors in order of importance:
  - Inline styles
  - id,
  - class, attribute, pseudo-class
  - type, pseudo-element

Result:



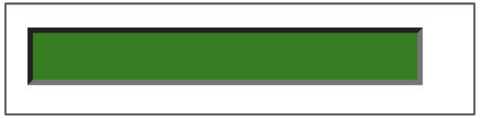
```css
.input-class {
    background-color: yellow;
}
input[type="email"] {
    background-color: green;
}
input {
    background-color: blue;
}
```

```html
<input type="email" class="input-class"
id="input-id" />
```

CSS Specificity

- Rules that determine which CSS values are applied in case of conflict.
- Selectors in order of importance:
    - Inline styles
    - id,
    - class, attribute, pseudo-class
    - type, pseudo-element

Result:



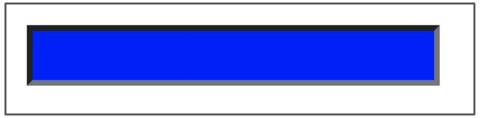
```
.container .input-class {
    background-color: green;
}
.input-class {
    background-color: yellow;
}
```

```
<div class="container" id="form">
    <input type="email" class="input-class"
id="input-id" />
</div>
```

CSS Specificity

- Rules that determine which CSS values are applied in case of conflict.
- Selectors in order of importance:
  - Inline styles
  - id,
  - class, attribute, pseudo-class
  - type, pseudo-element

Result:



```
#form .input-class {
    background-color: blue;
}
.container .input-class {
    background-color: green;
}
.input-class {
    background-color: yellow;
}
```

```
<div class="container" id="form">
    <input type="email" class="input-class"
id="input-id" />
</div>
```

Review

In the below code, which of the following is an HTML **tag**? Select all that apply.

```
<p class="intro">All About Trees</p>
<img src="tree.png" alt="A palm tree"/>


a) p
b) class
c) img
d) src
e) All About Trees
f) alt
```

In the below code, which of the following is an HTML **attribute**? Select all that apply.

```
<p class="intro">All About Trees</p>
<img src="tree.png" alt="A palm tree"/>


a) p
b) class
c) img
d) src
e) All About Trees
f) alt
```

What’s the difference between an **id** attribute and a **class** attribute?

Which CSS property creates spacing **outside** an element?

Which of the following is a **block** element? Select all that apply.

```
a) <section>

b) <header>

c) <div>

d) <a>

e) <img>

f) <input>
```

What CSS property can we use to **horizontally center** a block element?

```
a) padding

b) margin

c) text-align

d) none of the above
```