

# Web Development

## Bsc Applied, Forensics, Entertainment Systems, IOT

---

Eamonn de Leastar ([edeleastar@wit.ie](mailto:edeleastar@wit.ie))  
Dr. Brenda Mullally ([bmullally@wit.ie](mailto:bmullally@wit.ie))

Department of Computing, Maths &  
Physics

Waterford Institute of Technology

<http://www.wit.ie>

<http://elearning.wit.ie>



Waterford Institute of Technology  
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

# CSS - Part 2

---

Department of Computing, Maths &  
Physics

Waterford Institute of Technology

<http://www.wit.ie>

<http://elearning.wit.ie>



Waterford Institute of Technology  
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

# Agenda

- CSS

- Combining Rules & Selectors

- Classes and class based styling

- Selector rules and the “Cascade” in CSS

# Multiple Rules

```
h1 {  
    font-family: sans-serif;  
    color:      gray;  
}
```

```
h2 {  
    font-family: sans-serif;  
    color:      gray;  
}
```

```
p {  
    color: maroon;  
}
```

Here's the rule to select `<h1>` elements and change the font-family to sans-serif and the font color to gray. We'll talk a lot more about fonts later.

And here's another rule to do the exact same thing to the `<h2>` element.

# Combining Selectors (1)

- Rules can be combined if they are identical

```
h1 {  
    font-family: sans-serif;  
    color:      gray;  
}  
  
h2 {  
    font-family: sans-serif;  
    color:      gray;  
}  
  
p {  
    color: maroon;  
}
```

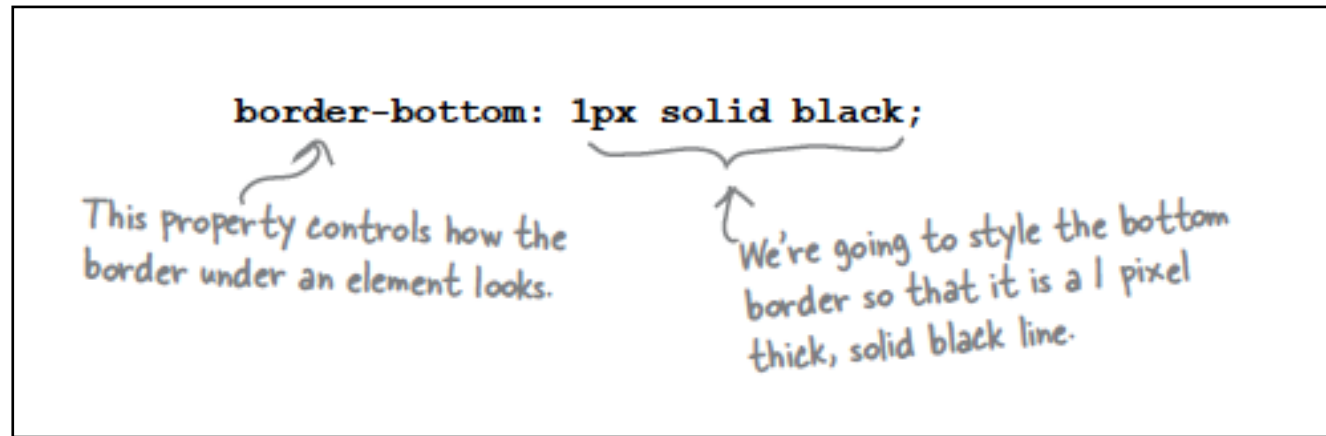
Here's the rule to select <h1> elements and change the font-family to sans-serif and the font color to gray. We'll talk a lot more about fonts later.

And here's another rule to do the exact same thing to the <h2> element.

```
h1, h2 {  
    font-family: sans-serif;  
    color:      gray;  
}
```

To write a rule for more than one element, just put commas between the selectors, like "h1, h2".

```
p {  
    color: maroon;  
}
```



- Placing the above rule associated with h1 “selector”, will draw a line - 1 pixel wide - under the heading in our site (you did this in lab1)

## Mobile Applications

---

### 1. [Apps](#)

# Combining Selectors (2)

```
h1, h2 {  
  font-family: sans-serif;  
  color: gray;  
}
```

The first rule stays the same. We're still going to use a combined rule for the font-family and color for both <h1> and <h2>.

```
h1 {  
  border-bottom: 1px solid black;  
}
```


But now we're adding a second rule that adds another property just to <h1>: the border-bottom property.

```
p {  
  color: maroon;  
}
```


- Both h1 and h2 share the font-family and color attributes, however only h1 is underlined

# Combining Rules

- Rules can be combined. The following two sets of style rules would produce identical results
- Rules can be listed separately:
- Or, rules can be grouped. Property:Value pairs need to be separated by a semicolon.



```
p {color: black;}  
p {background-color: teal;}  
p {padding: 1em;}  
p {margin: 1em;}  
p {font-family: helvetica, sans-serif;}  
p {text-align: justify;}
```



```
p  
{  
    color: black;  
    background-color: teal;  
    padding: 1em;  
    margin: 1em;  
    font-family: helvetica, sans-serif;  
    text-align: justify;  
}
```



# Combining Selectors

- Selectors can be combined into comma-separated groups.
- We combine the selectors so that a single declaration applies to multiple selectors.

```
h1 { color: maroon; }  
h2 { color: maroon; }  
h3 { color: maroon; }  
h4 { color: maroon; }  
h5 { color: maroon; }  
h6 { color: maroon; }
```

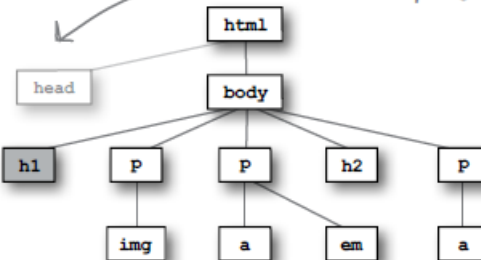
```
h1, h2, h3, h4, h5, h6 { color: maroon; }
```

# Seeing selectors visually

Let's take some selectors and see how they map to the tree you just created. Here's how this "h1" selector maps to the graph:

```
h1 {  
  font-family: sans-serif;  
}
```

This selector matches any <h1> elements in the page, and there's only one.

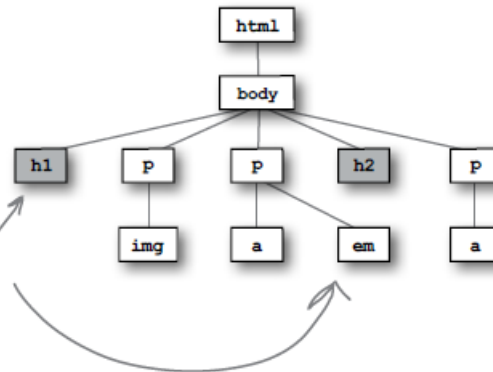


We can only style elements in the body, so we're not showing the <head> element and everything under it.

And here's how the "h1, h2" selector looks:

```
h1, h2 {  
  font-family: sans-serif;  
}
```

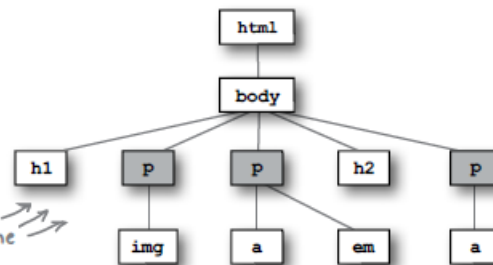
Now the selector matches both <h1> and <h2> elements.



If we use a "p" selector, here's how that looks:

```
p {  
  font-family: sans-serif;  
}
```

This selector matches all the <p> elements in the tree.



# Agenda

- CSS
  - Combining Rules & Selectors
  - Classes and class based styling
  - Selector rules and the “Cascade” in CSS

```

<body>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras sollicitudin, orci
    nec facilisis vehicula, neque urna porta risus, ut sagittis enim velit at orci.
  </p>
  <p>
    Fusce velit. Integer sapien enim, rhoncus vitae, cursus non,
    commodo vitae, felis. Nulla convallis ante sit amet urna. Maecenas condimentum
    hendrerit turpis.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras
    sollicitudin, orci nec facilisis vehicula, neque urna porta risus, ut sagittis enim
    velit at orci.
  </p>
  <p>
    Lorem ipsum dolor sit amet,<span>consectetur adipiscing elit</span>. Cras
    sollicitudin, orci
    nec acilisis vehicula, neque urna porta risus, ut sagittis enim velit at orci.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras
    sollicitudin, orci nec facilisis vehicula, neque urna porta risus, ut sagittis enim
    velit at orci.
  </p>
</body>

```

- How to style these paragraphs differently?
- Just using p as the selector will set the style for them all.

```

p
{
  color: black;
  background-color: teal;
}

```

# Class of elements

- Using a class of elements for styling
- You can then write a css rule to style any elements that belong to the class.
- Elements can be in more than one class

# Using **class** to identify elements

```
<body>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras sollicitudin, orci
    nec facilisis vehicula, neque urna porta risus, ut sagittis enim velit at orci.
  </p>
  <p class="withstyle">
    Fusce velit. Integer sapien enim, rhoncus vitae, cursus non,
    commodo vitae, felis. Nulla convallis ante sit amet urna. Maecenas condimentum
    hendrerit turpis.
  </p>
  <p class="warning">
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras
    sollicitudin, orci nec facilisis vehicula, neque urna porta risus, ut sagittis enim
    velit at orci.
  </p>
  <p>
    Lorem ipsum dolor sit amet, <span class="warning">consectetur adipiscing elit</span>.
    Cras sollicitudin, orci nec acilisis vehicula, neque urna porta risus, ut sagittis
    enim velit at orci.
  </p>
</body>
```

- To indicate that an element is a member of a class we use the class attribute.
- While the name of an element specifies its *type*, the class attribute lets you assign to it one or more *subtypes*.

```

<body>
<p>
  Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras sollicitudin, orci
  nec facilisis vehicula, neque urna porta risus, ut sagittis enim velit at orci.
</p>
<p class="withstyle">
  Fusce velit. Integer sapien enim, rhoncus vitae, cursus non,
  commodo vitae, felis. Nulla convallis ante sit amet urna. Maecenas
  condimentum
  hendrerit turpis.
</p>
<p class="warning">
  Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras
  sollicitudin, orci nec facilisis vehicula, neque urna porta risus, ut sagittis enim
  velit at orci.
</p>
<p>
  Lorem ipsum dolor sit amet,<span class="warning">consectetur adipiscing
  elit</span>.
  Cras sollicitudin, orci nec acilisis vehicula, neque urna porta risus, ut sagittis
  enim velit at orci.
</p>
</body>

```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras sollicitudin, orci nec facilisis vehicula, neque urna porta risus, ut sagittis enim velit at orci.

Fusce velit. Integer sapien enim, rhoncus vitae, cursus non, commodo vitae, felis. Nulla convallis ante sit amet urna. Maecenas condimentum hendrerit turpis.

**Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras sollicitudin, orci nec facilisis vehicula, neque urna porta risus, ut sagittis enim velit at orci.**

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras sollicitudin, orci nec facilisis vehicula, neque urna porta risus, ut sagittis enim velit at orci.

# Using Classes in CSS

- Class names are referenced in CSS as
  - element.classname

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras sollicitudin, orci nec facilisis vehicula, neque urna porta risus, ut sagittis enim velit at orci.

Fusce velit. Integer sapien enim, rhoncus vitae, cursus non, commodo vitae, felis. Nulla convallis ante sit amet urna. Maecenas condimentum hendrerit turpis.

**Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras sollicitudin, orci nec facilisis vehicula, neque urna porta risus, ut sagittis enim velit at orci.**

```
p
{
    background-color: white;
    color: black;
    font-family: times;
    margin: 0.5em;
    padding: 0.5em;
}

p.withstyle
{
    background-color: olive;
    color: navy;
    font-family: sans-serif;
    margin: 0.5em;
    padding: 0.5em;
}

p.warning
{
    background-color: yellow;
    color: red;
    font-weight: bold;
}
```



# Classes Independent of Elements

- May not have an element name preceding the period:

.classname

- Selector now matches *any* element of the given class

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras sollicitudin, orci nec facilisis vehicula, neque urna porta risus, ut sagittis enim velit at orci.

Fusce velit. Integer sapien enim, rhoncus vitae, cursus non, commodo vitae, felis. Nulla convallis ante sit amet urna. Maecenas condimentum hendrerit turpis.

**Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras sollicitudin, orci nec facilisis vehicula, neque urna porta risus, ut sagittis enim velit at orci.**

Lorem ipsum dolor sit amet, **consectetur adipiscing elit**. Cras sollicitudin, orci nec acilisis vehicula, neque urna porta risus, ut sagittis enim velit at orci.

```
p
{
  background-color: white;
  color: black;
  font-family: times;
  margin: 0.5em;
  padding: 0.5em;
}
```

```
.withstyle
{
  background-color: olive;
  color: navy;
  font-family: sans-serif;
  margin: 0.5em;
  padding: 0.5em;
}
```

```
.warning
{
  background-color: yellow;
  color: red;
  font-weight: bold;
}
```

# Agenda

- CSS
  - Combining Rules & Selectors
  - Classes and class based styling
  - Selector rules and the “Cascade” in CSS



Say you have an `<em>` element inside a paragraph. If you change the background color of the paragraph, do you think you also have to change the background of the `<em>` element so it matches the background color of the paragraph?

## The elements inside the `<p>` element inherit the font-family style from `<p>`

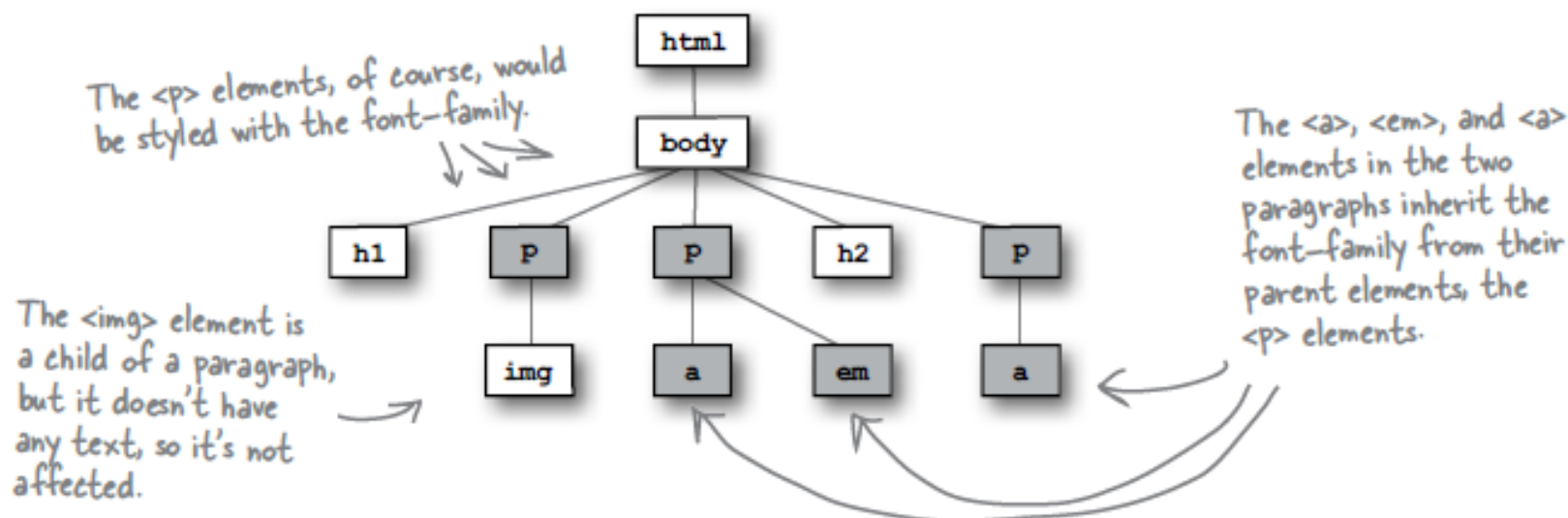
Just like you can inherit your blue eyes or brown hair from your parents, elements can inherit styles from their parents. In this case, the `<a>` and `<em>` elements inherited the font-family style from the `<p>` element, which is their parent element. It makes sense that changing your paragraph style would change the style of the elements in the paragraph, doesn't it? After all, if it didn't, you'd have to go in and add CSS rules for every inline element in every paragraph in your whole site...which would definitely be so NOT fun.

Not every style is inherited. Just some are, like font-family.

Not to mention, error-prone, tedious, and time-consuming

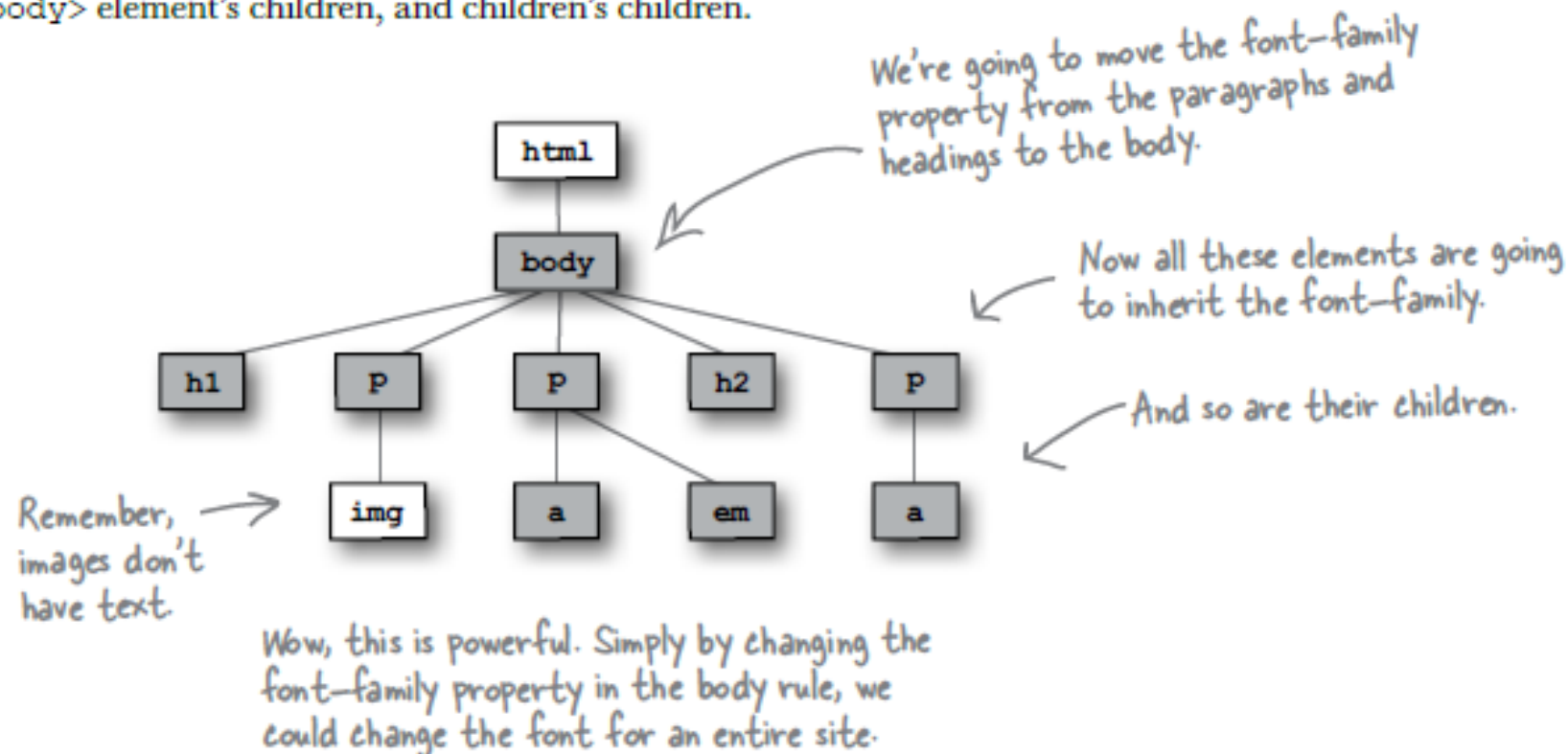
Let's take a look at the HTML tree to see how inheritance works:

If we set the font-family of all the `<p>` elements, here are all the elements that would be affected.



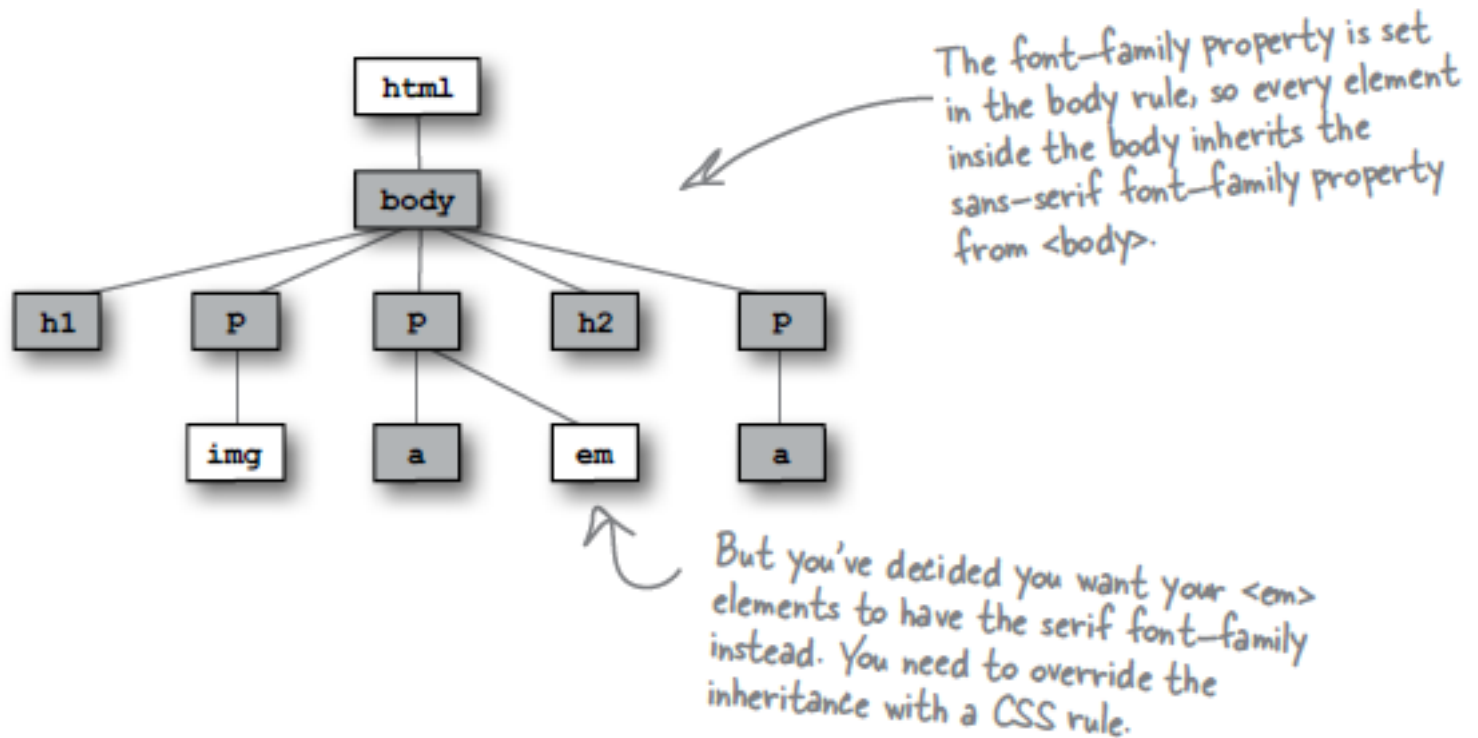
# What if we move the font up the family tree?

If most elements inherit the `font-family` property, what if we move it up to the `<body>` element? That should have the effect of changing the font for all the `<body>` element's children, and children's children.



# Overriding inheritance

By moving the `font-family` property up into the `body`, you've set that font style for the entire page. But what if you don't want the sans-serif font on every element? For instance, you could decide that you want `<em>` elements to use the serif font instead.



# Rules, Classes, Elements - which ones get selected?

(1) *Explicit Match*: Do any selectors select your element?

- Examine CSS rules for *explicit match* for element.

(2) *Inheritance Match*: What if no rules match the element:

- Rely on *inheritance*.
- Look at the element's parents, and parents' parents, and so on, until you find the property defined.

(3) *Default Match*: still no explicit or inherited match

- use the *default* value defined by the browser

(4) *Most Specific Match*: What if more than one match?

?

- select rule that is the *most specific*

# Examples

```
p { color: black; }
```

Here's a rule that selects any old paragraph element.

```
.greentea { color: green; }
```

This rule selects members of the greentea class. That's a little more specific.

```
p.greentea { color: green; }
```

And this rule selects only paragraphs that are in the greentea class, so that's even more specific.

```
p.raspberry { color: blue; }
```

```
p.blueberry { color: purple; }
```

These rules also select only paragraphs in a particular class. So they are about the same in specificity as the p.greentea rule.



# Specific & Ordering Match

- If we had an element that belonged only to the greentea class
  - there would be an obvious winner:
  - the p.greentea selector is the most specific, so the text would be green.
- But you have an element that belongs to all three classes: greentea, raspberry, and blueberry.
  - So, p.greentea, p.raspberry, and p.blueberry all select the element, and are of equal specificity. What do you do now?
  - You choose the one that is listed last in the CSS file.

(5): *Ordering Match*: - If you can't resolve a conflict because two selectors are equally specific

- use the ordering of the rules in your style sheet file. That is, you use the rule listed last in the CSS file (nearest the bottom). And in this case, that would be the p.blueberry rule.

# Example

```
p { color: black; }  
.greentea { color: green; }  
p.greentea { color: green; }  
p.raspberry { color: blue; }  
p.blueberry { color: purple; }
```

## 1- Explicit Match

## 2- Inheritance Match

## 3- Default Match

## 4- Most Specific Match

## 5- Ordering Match

```
<p>  
  My Normal Tea <br>  
  Customers say they <q>really like</q> this one!  
</p>  
<blockquote>  
  All of the best teas  
</blockquote>  
<p class="greentea">  
  My Green Tea  
</p>  
<p class="greentea blueberry">  
  My Mixed Tea - what colour is it?  
</p>
```

```
p      { color: black; }  
.greentea { color: green; }  
p.greentea { color: green; }  
p.raspberry { color: blue; }  
p.blueberry { color: purple; }
```

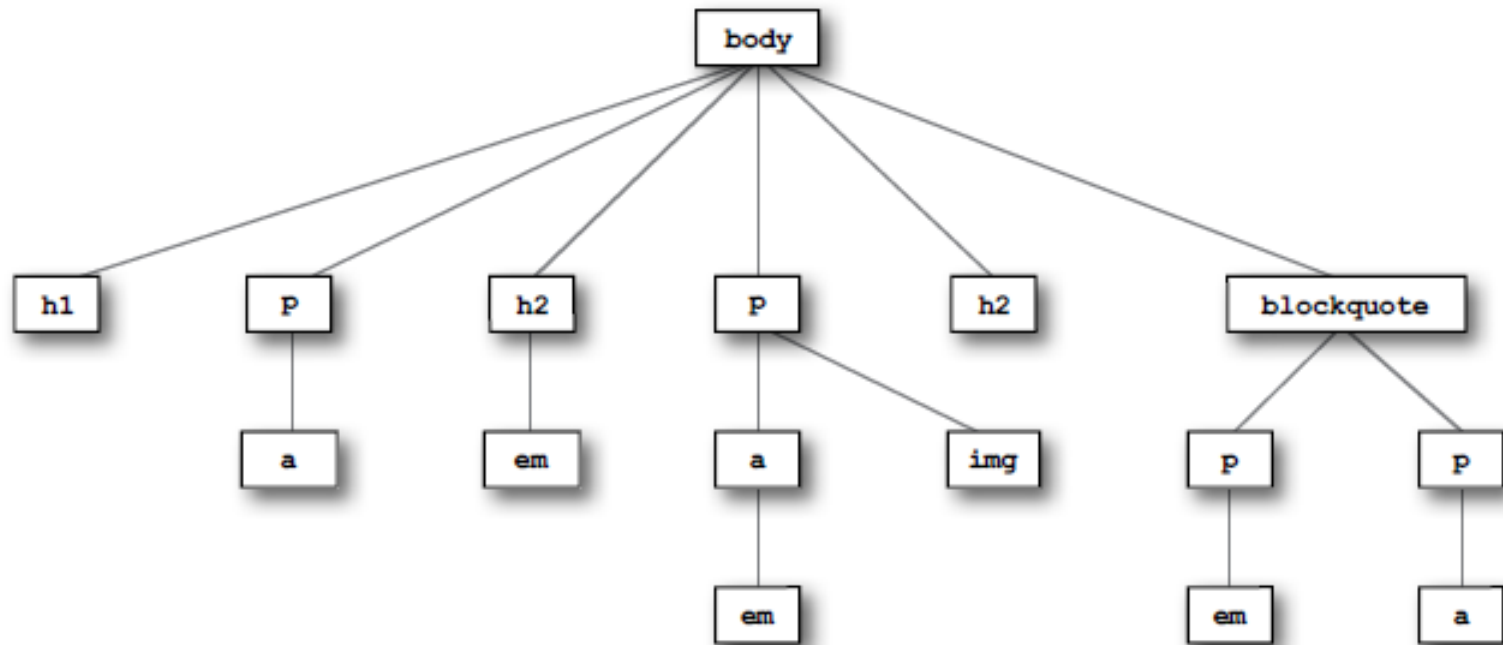
```
<p>  
  My Normal Tea <br>  
  Customers say they <q>really like</q> this one!  
</p>  
<blockquote>  
  All of the best teas  
</blockquote>  
<p class="greentea">  
  My Green Tea  
</p>  
<p class="greentea blueberry">  
  My Mixed Tea - what colour is it?  
</p>
```

My Normal Tea  
Customers say they "really like" this one!

All of the best teas

My Green Tea

My Mixed Tea - what colour is it?



```
body {  
    color: green;  
}  
  
p {  
    color: black;  
}
```

← Here's the CSS. Use this to determine which of the above elements hit the jackpot and get the green (color).

# Learning Outcomes

- On completion of this class you should:
  - Understand the motivations for using CSS in web site development
  - Be able to compose simple CSS rules and incorporate them into a separate CSS file for a site.
  - Try the exercises in the book for chapter 7 and read pages 300-302