CIS 371 Web Application Programming

Styles in CSS

Lecturer: Dr. Yong Zhuang

Based on the original version by Professor Hans Dulimarta.

Review

- What is CSS and its history.
- How to apply CSS to HTML: External, Internal, Inline and their Cascading Order
- How to define styles
- CSS Box Model: Padding(inside border), Margin(outside border)
- CSS colors: 140 standard names, RGB, Hex string, HSL
- Font size: 1em: the width of the Uppercase “M” in the current font.
- CSS selector: id, tag name, class name, attribute.
- CSS selector Specificity.

(Block vs. Inline) & display

- Block-level elements: start on a new line and **take up the available full-width (of its parent)**
- Inline elements: do not start on a new line and **take up as much width as necessary**
- Change default behavior using **display**
  - display: inline => changes a block element to an inline element
  - display: block => changes an inline element to a block element
  - display: none => hide the element
- Examples

Box/Element Positioning

	Description	**Use top, bottom, left, right**
position: static	Default	No
position: relative	Apply adjustment from **its own default position**	Yes: adjustment amount
position: absolute	Apply adjustment **relative to nearest positioned parent**	Yes: adjustment amount
position: fixed	Absolute position within the browser viewport	Yes

Relative Positioning: adjusted from own default position

```
<!-- HTML -->
<p>Text in <span>default</span> position</p>
<p>Text in <span class=”sample”>adjusted</span> position</p>
```

Text in default position
Text in adjusted position

Relative Positioning: adjusted from own default position

```
<!-- HTML -->
<p>Text in <span>default</span> position</p>
<p>Text in <span class="sample">adjusted</span> position</p>
```

Apply adjustment from **its own default position**

```
/* CSS */
.sample {
    color: red;
    position: relative;
    top: 5px;
    right: 10px;
}
```

Text in default position
Text in adjusted position

1 minute…

Relative Positioning: adjusted from own default position

```
<!-- HTML -->
<p>Text in <span>default</span> position</p>
<p>Text in <span class="sample">adjusted</span> position</p>
```

```
/* CSS */
.sample {
    color: red;
    position: relative;
    top: 5px;
    right: 10px;
}
```

Text in default position
Text in adjusted position

Shifted 5 pixels (down) from the default top, 10 pixels (left) from the default right

Absolute Positioning: adjusted from nearest parent

```
<!-- HTML -->
<p>Text in <span>default</span> position</p>
<p>Text in <span class=”sample”>adjusted</span> position</p>
```

Text in default position
Text in adjusted position

Absolute Positioning: adjusted from nearest parent

```
<!-- HTML -->
<p>Text in <span>default</span> position</p>
<p>Text in <span class="sample">adjusted</span> position</p>
```

Apply adjustment **relative to nearest positioned parent**

```
/* CSS */
p {
    position: relative;
    border: 1px solid blue;
}
.sample {
    color: red;
    position: absolute;
    top: 5px;
    right: 10px;
}
```

Text in default position

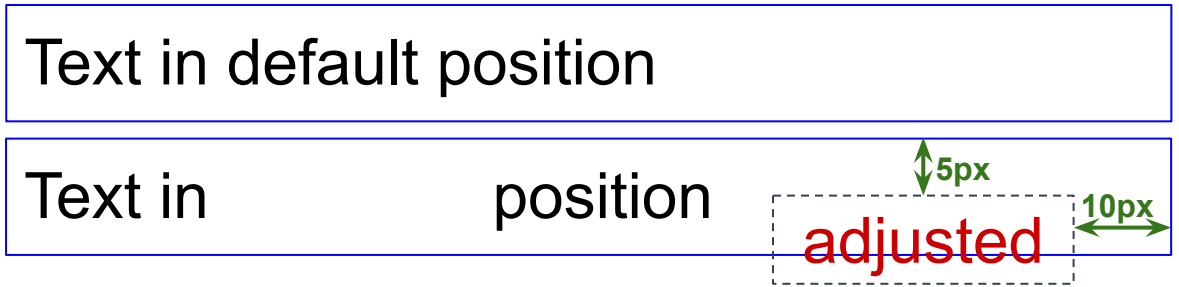
Text in adjusted position

1 minute…

Absolute Positioning: adjusted from nearest parent

```
<!-- HTML -->
<p>Text in <span>default</span> position</p>
<p>Text in <span class="sample">adjusted</span> position</p>
```

```
/* CSS */
p {
    position: relative;
    border: 1px solid blue;
}
.sample {
    color: red;
    position: absolute;
    top: 5px;
    right: 10px;
}
```



Shifted 5 pixels (down) from the parent top, 10 pixels (left) from the parent right

CSS Cheat sheet(s)

https://htmlcheatsheet.com/css/

https://devhints.io/css

CSS Selectors

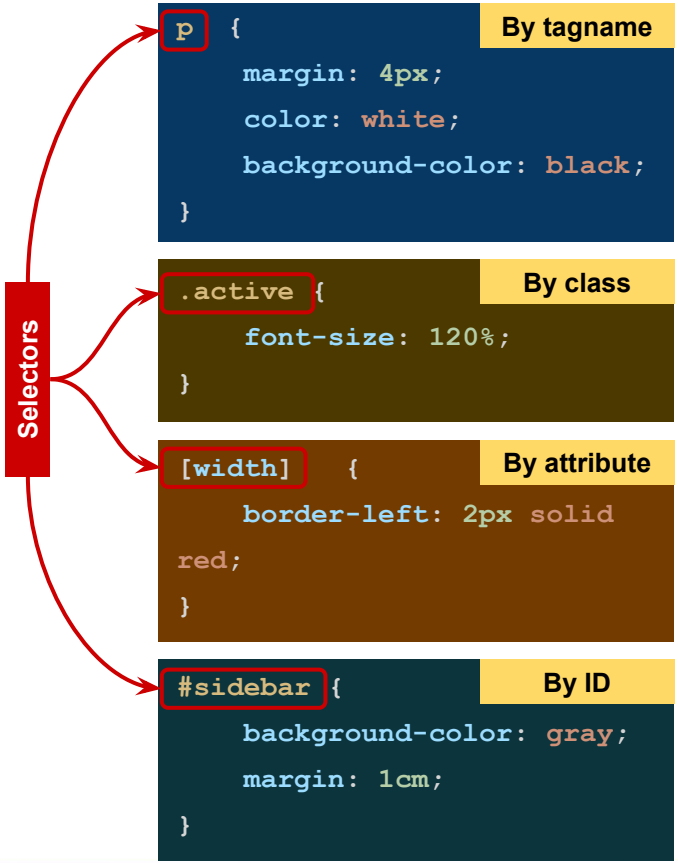
How to selectively apply styles to specific part(s) of the DOM tree?

Being selective / (more) specific

The company will hire

- A student
- A GVSU student
- A GVSU student graduated before 2022
- A GVSU student graduated before 2022 with GPA at least 3.2
- A GVSU marketing student graduated with GPA at least 3.2 before 2022
- … and so on

CSS Selectors



The 3-property rule applies to **any paragraph**

The 1-property rule applies to **any elements** with class “active”

The 1-property rule applies to **any elements** that include the **attribute** “width”

The 2-property rule applies to **only one element** with **id** “sidebar”

general

specific

CSS “selectors” / “filters”

Various options to select portions(s) of the DOM tree. Select by:

- ID, tag name, CSS class (or combination of them)
- Attribute (with or without its value)
- Parent/Child relationship in the DOM tree, such as
  - All immediate children of ____
  - Any descendants of _____
  - The last grandchild of _____
  - and so on…
- Sibling relationship in the DOM tree
- Permutations of all the above selectors

CSS “selectors” / “filters”

A CSS selector targets DOM elements

(it does NOT target text nodes)

Selector Permutations: tag & class

```
/* in CSS */
li.fruit {
    color: red
}
```

Apply only to list items with class .fruit

1. Strawberry
2. Raspberry Pi
3. Halle Berry

```
<!-- in HTML -->
<ol>
    <li class="fruit">Strawberry</li>
    <li class="device">RaspBerry Pi</li>
    <li>Halle Berry</li>
</ol>
```

Selector Permutations: tag & class

```
/* in CSS */
li.fruit {
    color: red
}
```

Apply only to list items with class .fruit

1. Strawberry
2. Raspberry Pi
3. Halle Berry

```
<!-- in HTML -->
<ol>
    <li class="fruit">Strawberry</li>
    <li class="device">RaspBerry Pi</li>
    <li>Halle Berry</li>
</ol>
```

```
/* in CSS */
li .fruit {
    color: red
}
```

Beware of SPACE.

This rule applies to **descendants** of
NOT <li> themselves.

1. Strawberry
2. Raspberry Pi
3. Halle Berry

Selector Permutations: tag & attribute

```
/* in CSS */
li[class] {
      color: red
}
```

Apply only to list items with **class** attribute set, regardless of its value

```
<!-- in HTML -->
<ol>
    <li class="fruit">Strawberry</li>
    <li class="device">RaspBerry Pi</li>
    <li>Halle Berry</li>
</ol>
```

Selector Permutations: tag & attribute

```
/* in CSS */
li[class] {
      color: red
}
```

Apply only to list items with **class** attribute set, regardless of its value

```
<!-- in HTML -->
<ol>
    <li class="fruit">Strawberry</li>
    <li class="device">RaspBerry Pi</li>
    <li>Halle Berry</li>
</ol>
```

1. Strawberry
2. Raspberry Pi
3. Halle Berry

Selector Permutations: tag & attr & attr-value

```
/* in CSS */
li[class*=t] {
      color: red
}
```

Apply only to list items with **class** attribute value containing "**t**"

```
<!-- in HTML -->
<ol>
    <li class="fruit">Strawberry</li>
    <li class="device">RaspBerry Pi</li>
    <li class="actor">Halle Berry</li>
</ol>
```

Selector Permutations: tag & attr & attr-value

```
/* in CSS */
li[class*=t] {
      color: red
}
```

Apply only to list items with **class** attribute value containing "**t**"

```
<!-- in HTML -->
<ol>
    <li class="fruit">Strawberry</li>
    <li class="device">RaspBerry Pi</li>
    <li class="actor">Halle Berry</li>
</ol>
```

1. Strawberry
2. Raspberry Pi
3. Halle Berry

Use Cases

Apply the particular styles **only to**:

- Odd rows (or even rows) of a table
- Paragraph immediately after heading level 2
- Input fields for password
- Empty list items
- Bold text inside the last row of a table
- ....

Selectors: by attribute(s)

Objective: select elements with a particular attribute

Selectors

- [attr] ⇒ select elements that have attribute attr (regardless of its value)
- [attr=val] ⇒ select elements whose attribute attr is set to “val”
- [attr~=val] ⇒ select elements whose attribute attr **contains** “val” (whole word)
- [attr*=val] ⇒ select elements whose attribute attr **contains** “val” (partial word)
- [attr|=val] ⇒ select elements whose attribute attr **starts with** “val” (whole word)
- [attr^=val] ⇒ select elements whose attribute attr **starts with** “val” (partial word)
- [attr$=val] ⇒ select elements whose attribute attr **ends with** “val” (partial word)
- [attr1=val2][attr2*=val2] ⇒ use multiple attributes (**logical and**)

Selector by relative placement in DOM tree

Descendant/Younger Sibling Selectors

Types	Selector	Apply Rules to
Immediate children	`div > p { ...rules... }`	paragraphs which are an immediate children of div
Any descendant	`div p { ...rules... }`	paragraphs which are a descendant of a div (immediate children included)
Immediate (**younger**) sibling	`div + p { ...rules... }`	one paragraph (immediate younger sibling of a div)
Any **younger** sibling	`div ~ p { ...rules... }`	paragraphs which are younger sibling of a div (immediate sibling included)

Examples

Target Element in Complex Selectors

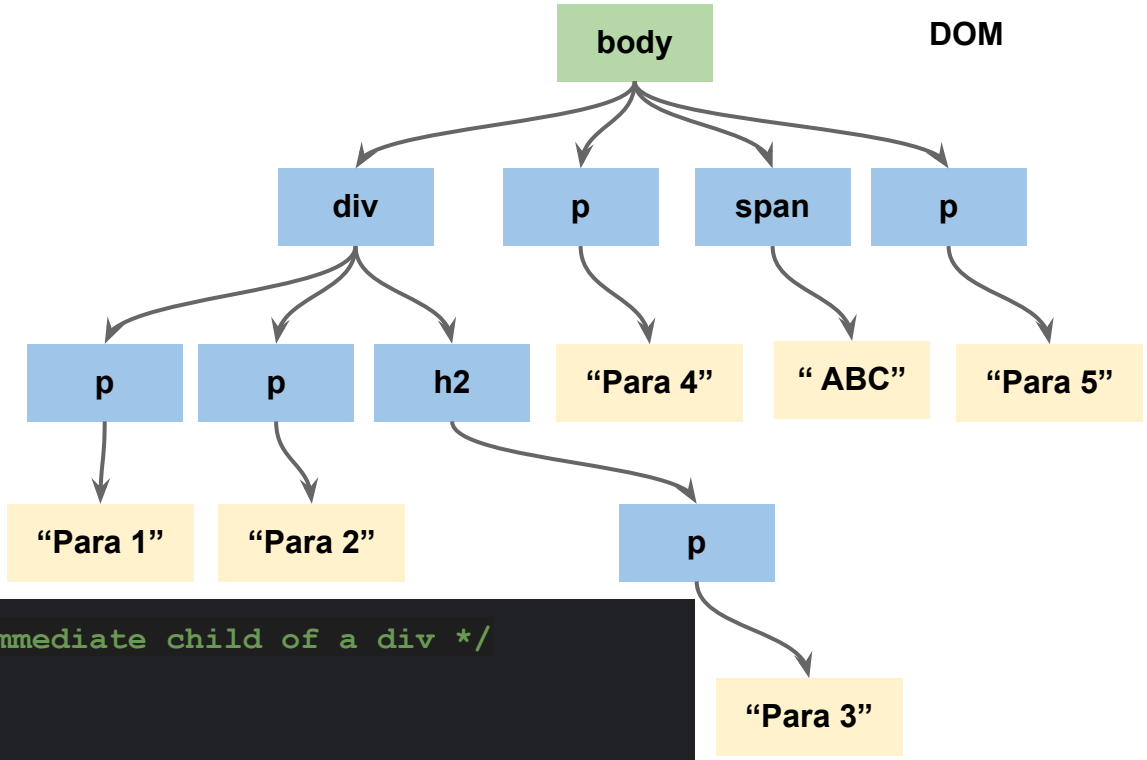
Selector	Target Element
`div > p`	
`h2 > p table`	
`h2 ~ p table`	
`#top div p .delay`	

Target Element in Complex Selectors

Selector	Target Element
`div > p`	Paragraph <p>
`h2 > p table`	Table <table>
`h2 ~ p table`	Table <table>
`#top div p .delay`	Elements with class .delay

Child (Immediate Descendant) Selector

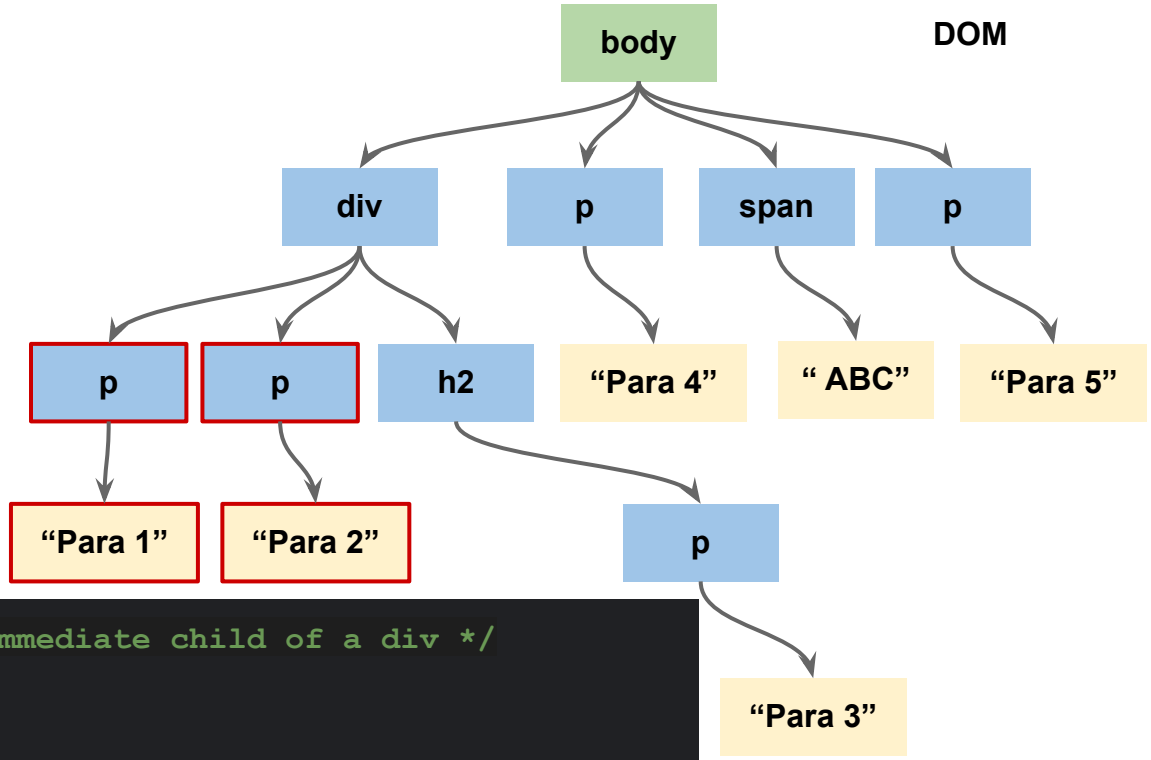
```
<body>
    <div>
        <p>Para 1</p>
        <p>Para 2</p>
        <h2><p>Para 3</p></h2>
    </div>
    <p>Para 4</p>
    <span>ABC</span>
    <p>Para 5</p>
</body>
```




```
/* apply to paragraphs which are an immediate child of a div */
div > p {
    border: 2px solid red;
}
```

Child (Immediate Descendant) Selector

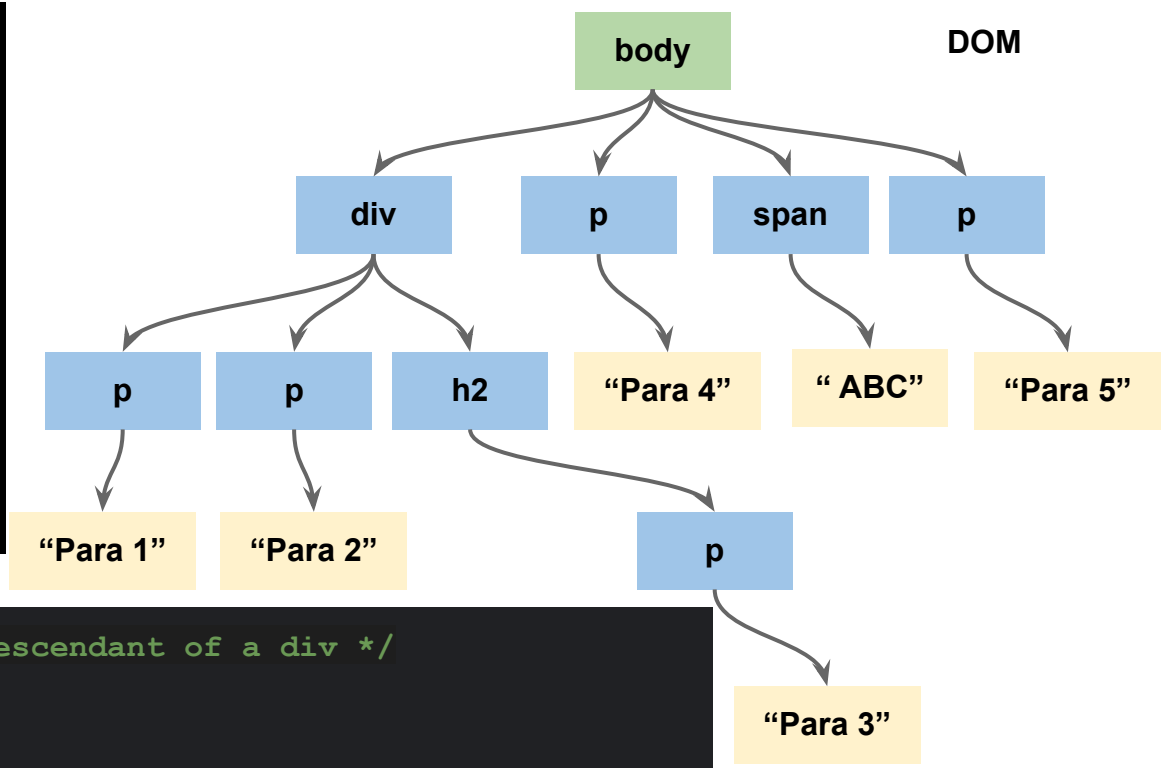
```
<body>
    <div>
        <p>Para 1</p>
        <p>Para 2</p>
        <h2><p>Para 3</p></h2>
    </div>
    <p>Para 4</p>
    <span>ABC</span>
    <p>Para 5</p>
</body>
```




```
/* apply to paragraphs which are an immediate child of a div */
div > p {
    border: 2px solid red;
}
```

(Deeper) Descendant Selector

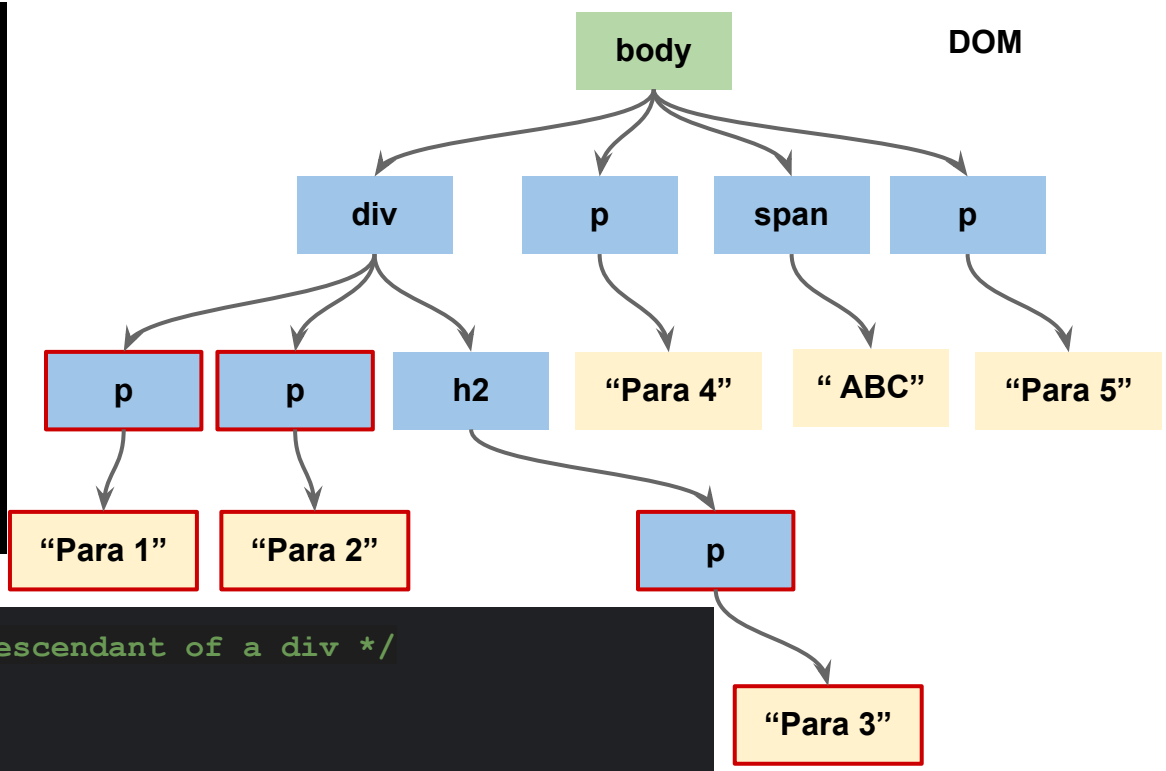
```
<body>
    <div>
        <p>Para 1</p>
        <p>Para 2</p>
        <h2><p>Para 3</p></h2>
    </div>
    <p>Para 4</p>
    <span>ABC</span>
    <p>Para 5</p>
</body>
```




```
/* apply to paragraphs which are a descendant of a div */
div  p {
    border: 2px solid red;
}
```

(Deeper) Descendant Selector

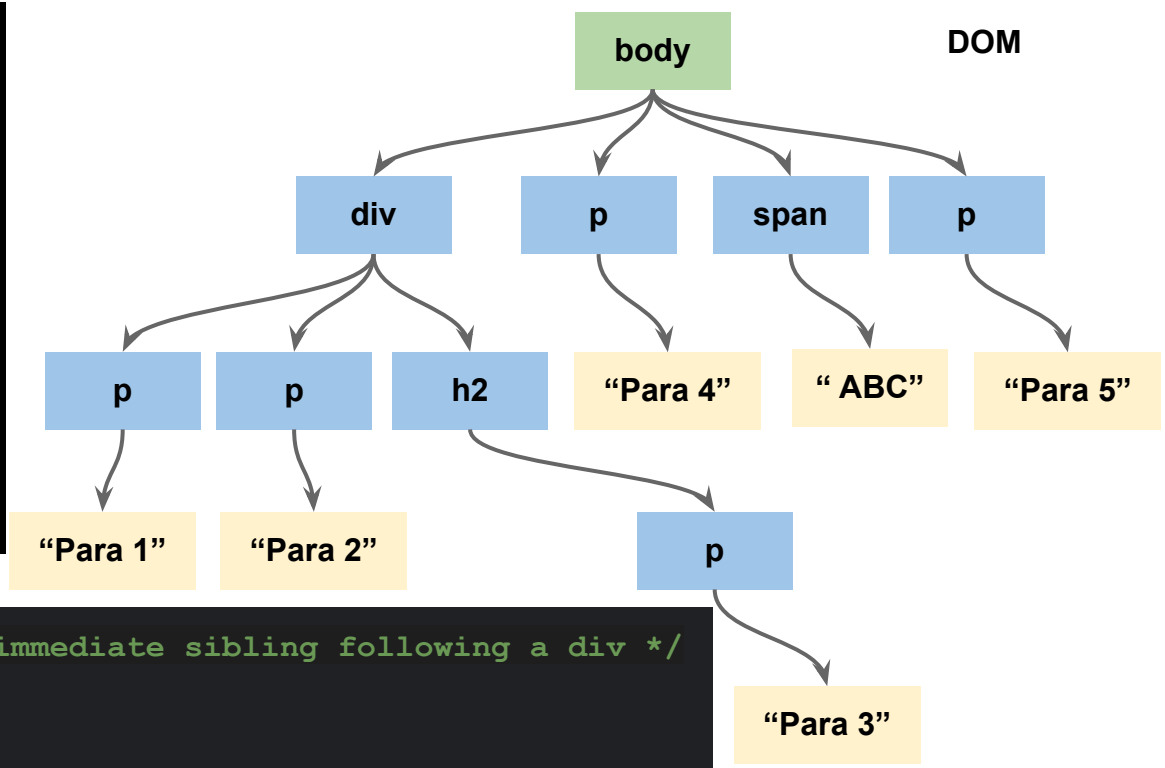
```
<body>
    <div>
        <p>Para 1</p>
        <p>Para 2</p>
        <h2><p>Para 3</p></h2>
    </div>
    <p>Para 4</p>
    <span>ABC</span>
    <p>Para 5</p>
</body>
```




```
/* apply to paragraphs which are a descendant of a div */
div  p {
    border: 2px solid red;
}
```

Immediate Sibling Selector

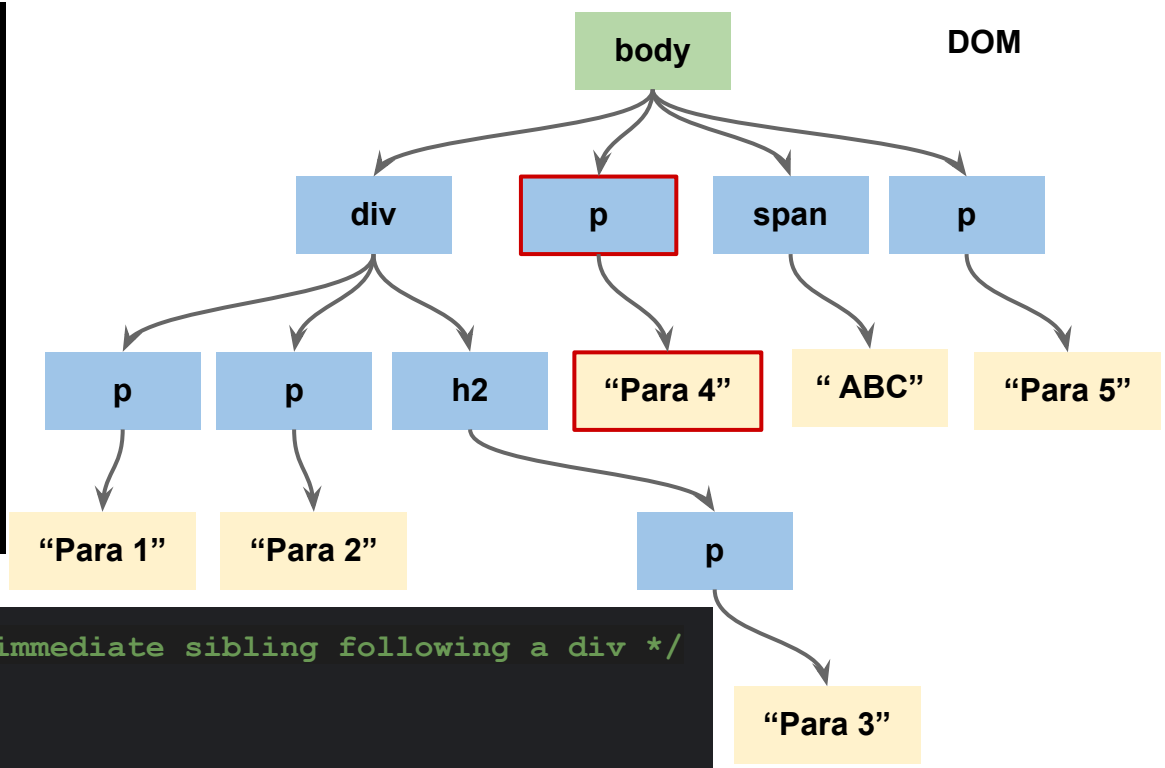
```
<body>
    <div>
        <p>Para 1</p>
        <p>Para 2</p>
        <h2><p>Para 3</p></h2>
    </div>
    <p>Para 4</p>
    <span>ABC</span>
    <p>Para 5</p>
</body>
```




```
/* apply to paragraphs which are an immediate sibling following a div */
div + p {
    border: 2px solid red;
}
```

Immediate Sibling Selector

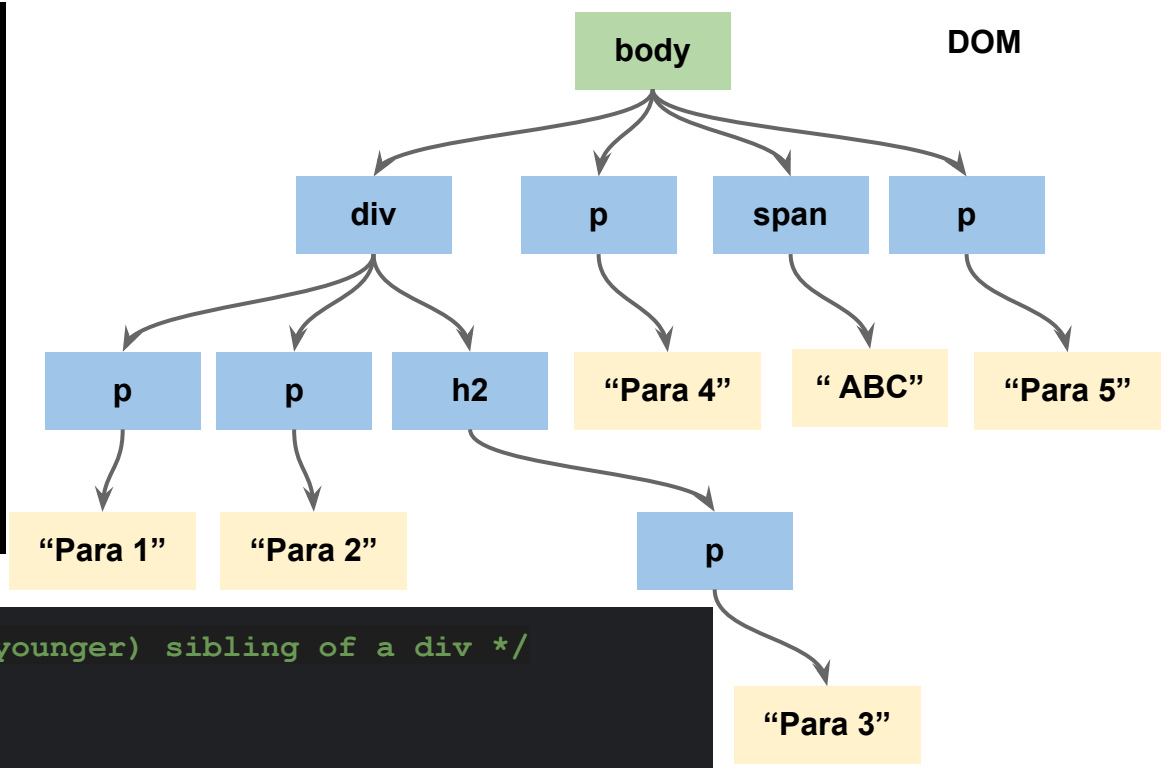
```
<body>
    <div>
        <p>Para 1</p>
        <p>Para 2</p>
        <h2><p>Para 3</p></h2>
    </div>
    <p>Para 4</p>
    <span>ABC</span>
    <p>Para 5</p>
</body>
```




```
/* apply to paragraphs which are an immediate sibling following a div */
div + p {
    border: 2px solid red;
}
```

General Siblings Selector

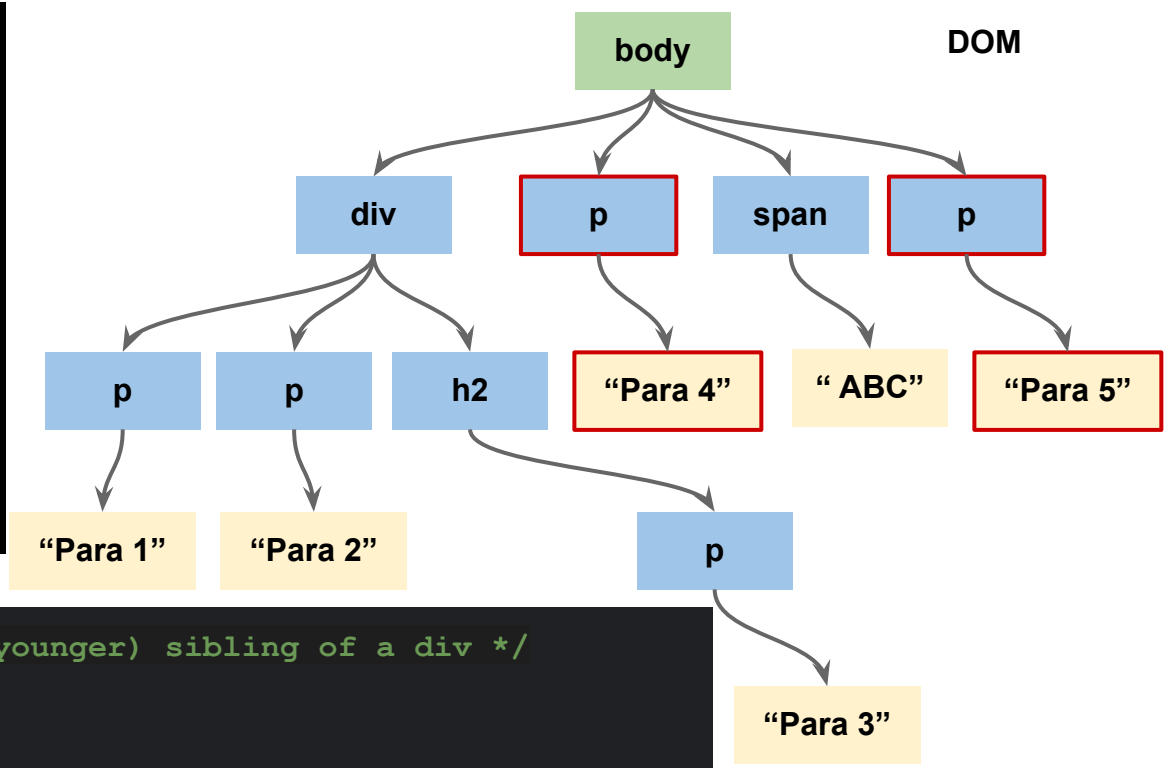
```
<body>
    <div>
        <p>Para 1</p>
        <p>Para 2</p>
        <h2><p>Para 3</p></h2>
    </div>
    <p>Para 4</p>
    <span>ABC</span>
    <p>Para 5</p>
</body>
```




```
/* apply to paragraphs which are a (younger) sibling of a div */
div ~ p {
    border: 2px solid red;
}
```

General Siblings Selector

```
<body>
    <div>
        <p>Para 1</p>
        <p>Para 2</p>
        <h2><p>Para 3</p></h2>
    </div>
    <p>Para 4</p>
    <span>ABC</span>
    <p>Para 5</p>
</body>
```




```
/* apply to paragraphs which are a (younger) sibling of a div */
div ~ p {
    border: 2px solid red;
}
```

Chain of descendant/siblings

When the selector has multiple "groups" of pattern, which group of elements will be impacted?

Selector & Rule(s)	Rule(s) applied to
`p  li span { font-size: 125% }`	
`div p ~ ol img { border: 2px dashed blue; }`	
`div + p > ol.discount table { width:100% }`	

Chain of descendant/siblings

When the selector has multiple “groups” of pattern, the CSS rules apply to the rightmost group

Selector & Rule(s)	Rule(s) applied to
`p li span { font-size: 125% }`	
`div p ~ ol img { border: 2px dashed blue; }`	<img>
`div + p > ol.discount table { width:100% }`	<table>

Selector Modifiers :pseudo-classes

- Links (:link, :visited, :hover, :active)
- Input (:checked, :disabled, :enabled, :focus, :in-range, :out-of-range, :invalid, :valid, :optional, :required, :read-only, :read-write)
- Child order (:first-child, :last-child, :nth-child, :nth-last-child, :only-child)
- Of-Type order (:first-of-type, :last-of-type, :nth-of-type, :nth-last-of-type, :only-of-type)
- Online reference (look for “Pseudo-classes” on the left)

:first-child vs. :first-of-type

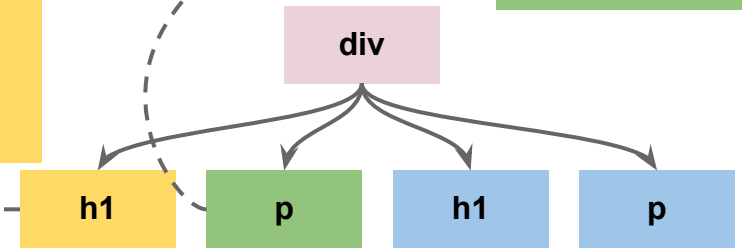
```
<div>
    <h1>First Heading</h1>
    <p>One paragraph</p>
    <h1>Second Heading</h1>
    <p>A bit longer paragraph</p>
</div>
```

```
div p:first-child {
    /* no matching element */
      color: red
}
```

The **first** “daugher” in a family may be the **third** “child”

```
div p:first-type {
    /* no matching element */
      color: red
}
```

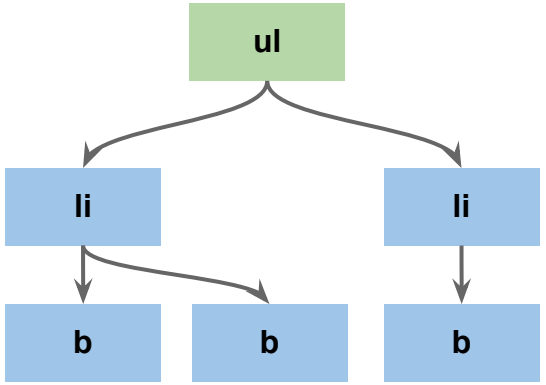
```
div h1:first-child {
    /* no matching element */
      color: red
}
```



:first-child

```
<ul>
    <li>Test <b>one</b> and <b>two</b></li>
    <li>Another <b>text</b></li>
</ul>
```

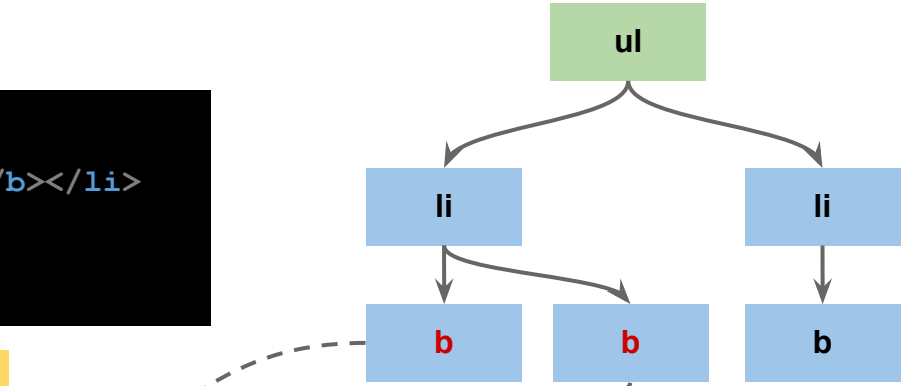
```
li:first-child b{
      color: red;
}
```



:first-child

```
<ul>
    <li>Test <b>one</b> and <b>two</b></li>
    <li>Another <b>text</b></li>
</ul>
```

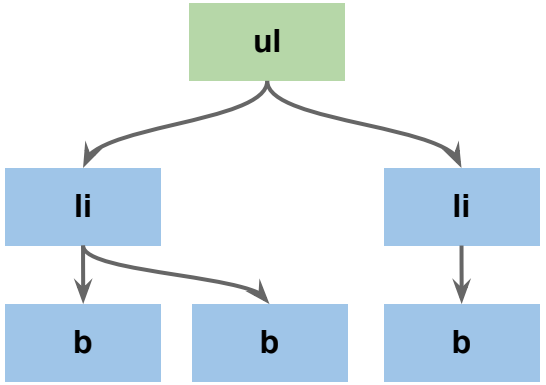
```
li:first-child b{
      color: red;
}
```



:first-child

```
<ul>
    <li>Test <b>one</b> and <b>two</b></li>
    <li>Another <b>text</b></li>
</ul>
```

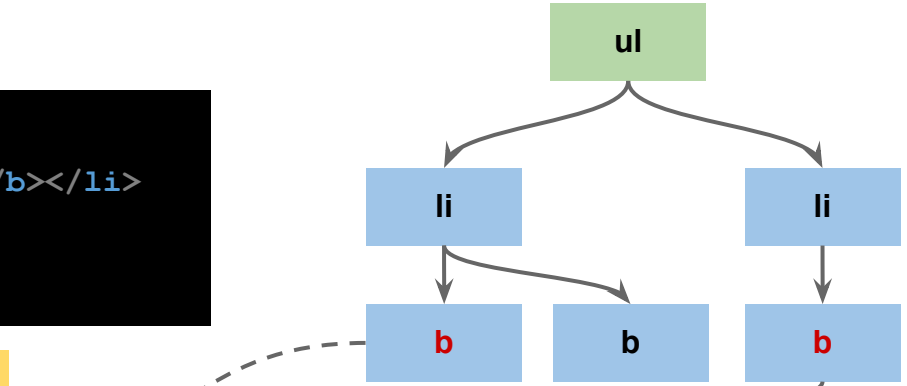
```
li b:first-child{
      color: red;
}
```



:first-child

```
<ul>
    <li>Test <b>one</b> and <b>two</b></li>
    <li>Another <b>text</b></li>
</ul>
```

```
li b:first-child{
      color: red;
}
```



CSS3 :nth-child()

- :nth-child(4): select child #4
- :nth-child(odd): select children #1, #3, #5, …
- :nth-child(even): select children #2, #4, #6, …
- :nth-child(3n+1): select children #1, #4, #7, #10, ...

CSS Pseudo Classes Example

bold when mouse is over text

```
/* stylesheet: white on green */
h1.active {
    color: white;
    background-color: green;
}
p.active:hover {
    font-weight: bold
}
```

First Heading
This text is terse.

Second Heading
This text is slightly longer than the previous one.

```
<!-- HTML doc -->
<h1>First Heading</h1>
<p class="active">This text is terse.</p>

<h1 class="active">Second Heading</h1>
<p>This text is slightly longer than the previous one.</p>
```

Pseudo Classes vs. Pseudo Elements

Pseudo Classes

- Practical use: **select only elements in a particular "state"**
- Link states (:link, :visited, :hover, :active)
- Input states (:checked, :disabled, :empty, :enabled, :focus)
- Positional (:first-child, :last-child, :nth-child(), :nth-last-child())

Pseudo Elements

- Practical use: **select only certain part of an element**
- Selectors ("::" in CSS3, ":" in CSS[1|2])
  - ::after
  - ::before
  - ::first-letter
  - ::first-line
  - ::selection

References: Many more Mozilla Dev Network

CSS3 Selectors

Interactive CSS Selectors at

https://www.w3schools.com/cssref/trysel.asp

Media Query: @media

Styles that apply when browser canvas width > 600px

Styles that apply when browser canvas width <= 600px

```
p {
    font-size: 110%;
}
.box {
    background: red;
}

@media (max-width: 600px) {
    p {
        font-size: 85%;
    }
    .box {
        background: blue;
    }
}
```

Other query params: (min|max)-height, *-resolution, *-aspect-ratio, *-color, and many more

What is the effect of the CSS Styles?

```
/* CSS */
div > ol {
    display: none;
}
div:hover ol {
    display: block;
}
```

```
<!-- HTML doc -->
<div>
    <div>Parent</div>
    <ol>
        <li>Child 1</li>
        <li>Child 2</li>
        <li>Child 3</li>
    </ol>
</div>
```