

---

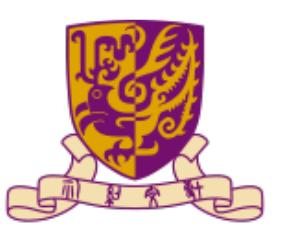
# **CSC4130**

# **Introduction to Human-Computer Interaction**

## **Lecture 6**

## **User Interface Technology: HTML and CSS**

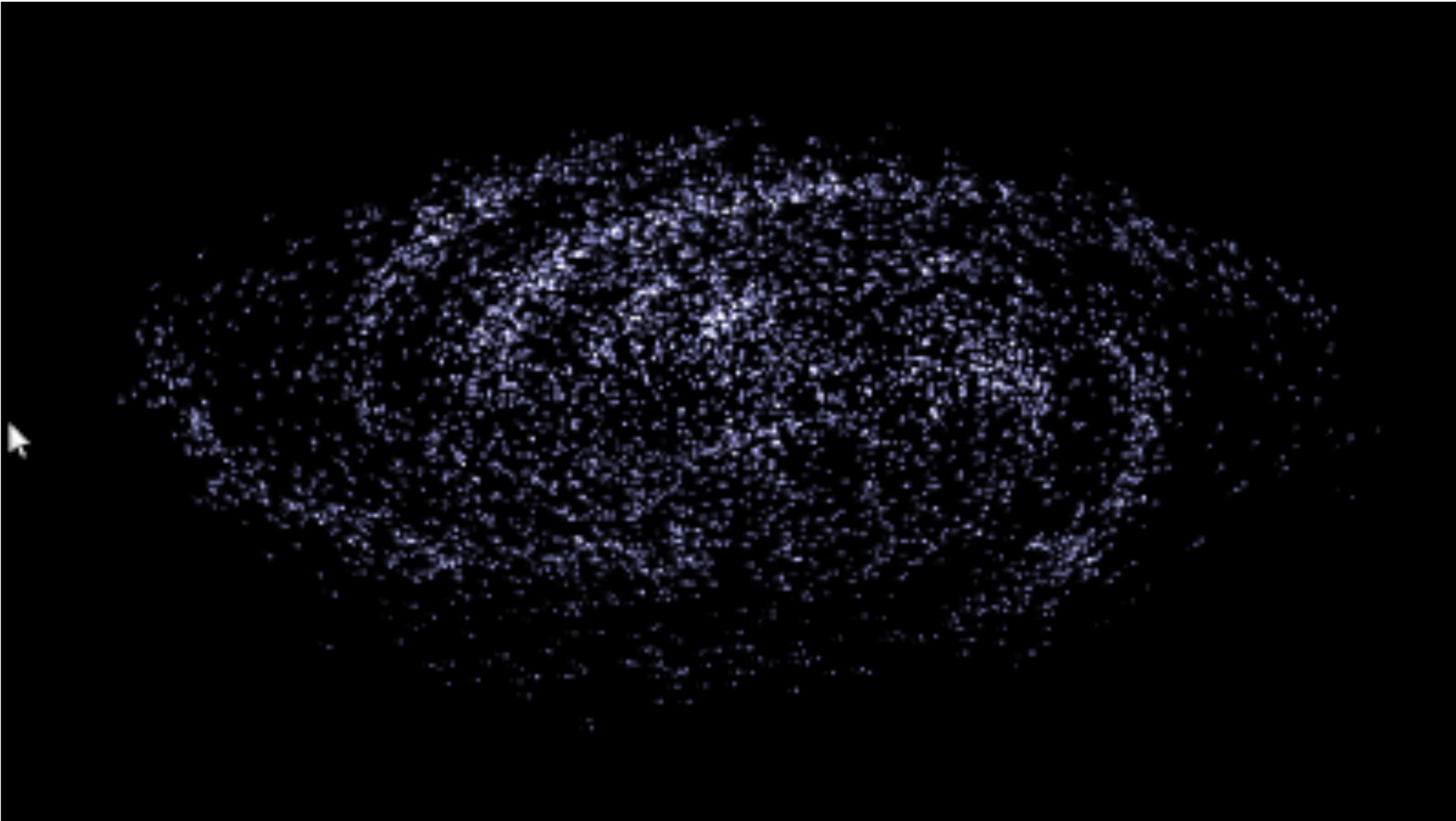




香港中文大學(深圳)

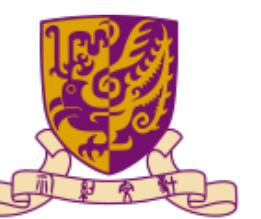
The Chinese University of Hong Kong, Shenzhen

# Simulation



# Game



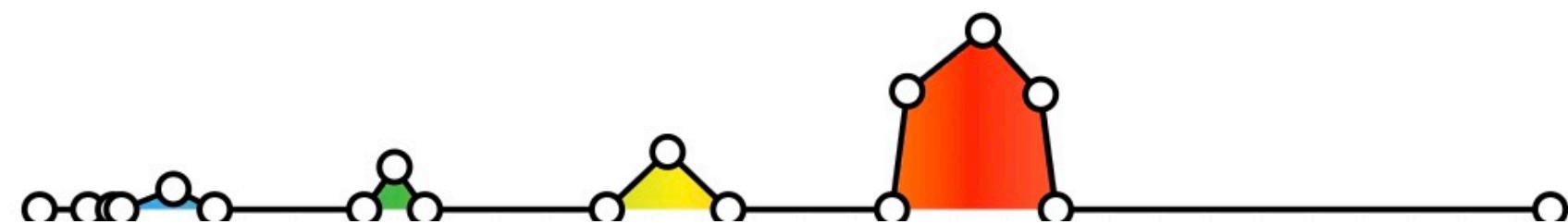
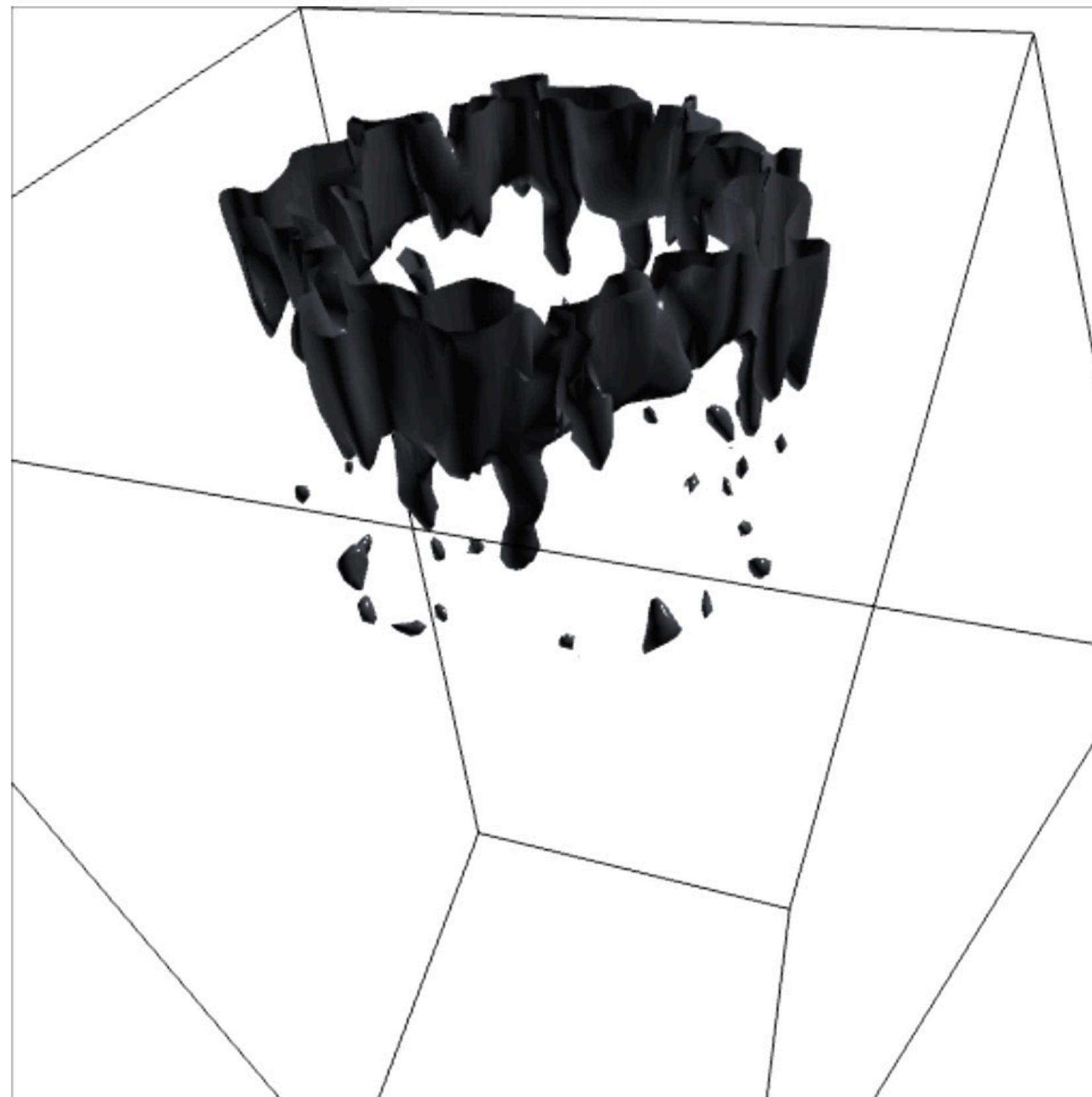
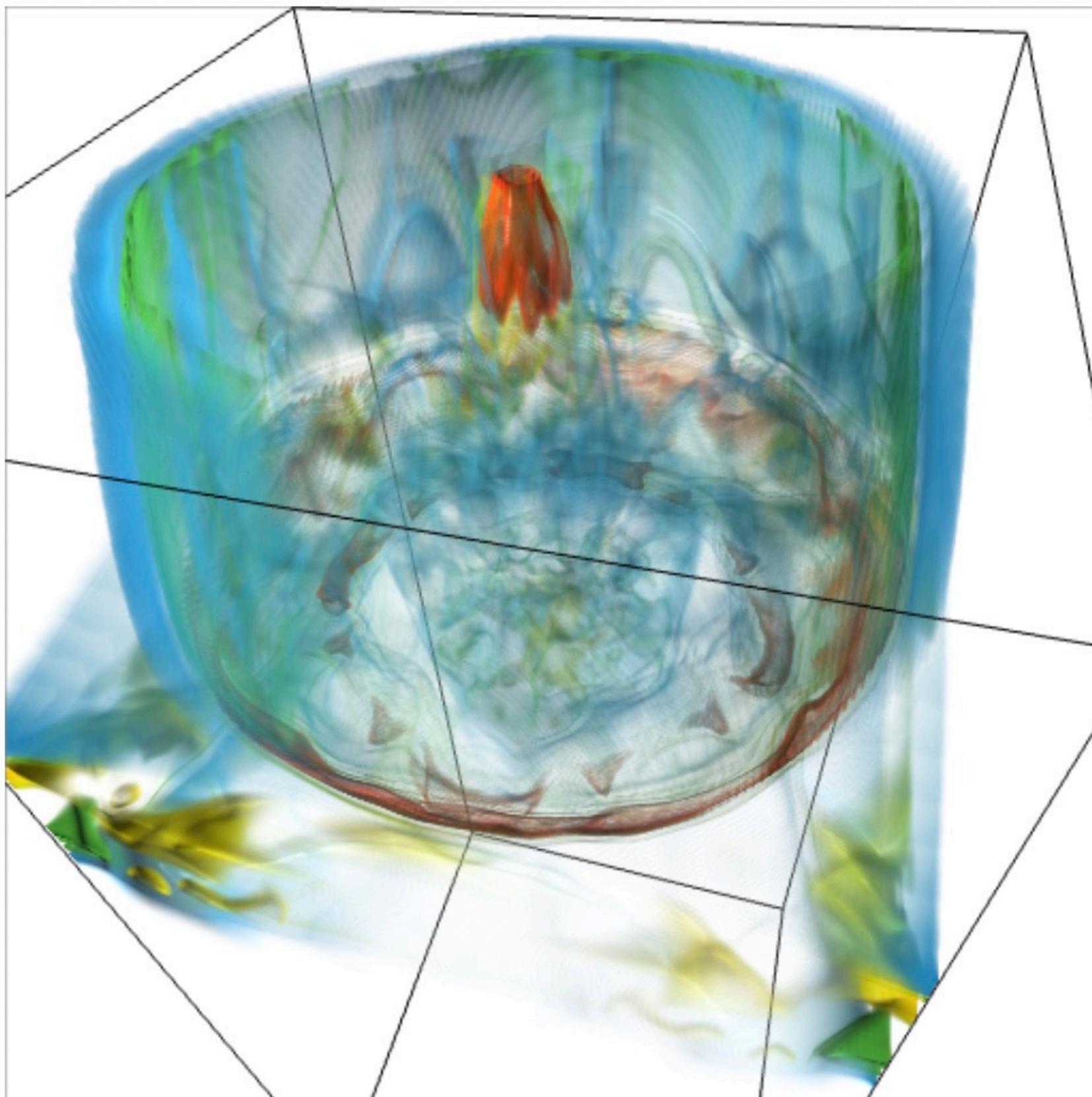


香港中文大學(深圳)

The Chinese University of Hong Kong, Shenzhen

# Visualization and interaction

## VolumeVisual



## Control Panel

[about](#)

[help?](#)

Ionization (PD) ▾

Cutting Plane ▾

Lighting Parameters ▾

Ambient  
0.0  1.0

Diffuse  
0.0  1.0

Specular  
0.0  1.0

Shininess  
2.0  128.0

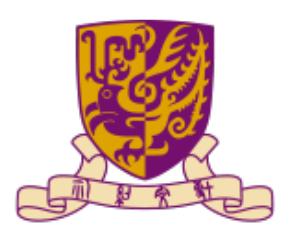
Contrast  
0.0  1.0

Viewing Parameters ▾

Volume Rendering ▾

# Virtual reality





香港中文大學(深圳)

The Chinese University of Hong Kong, Shenzhen

# Outline

- HTML
- CSS



香港中文大學(深圳)

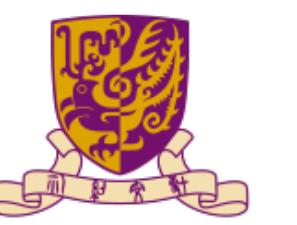
The Chinese University of Hong Kong, Shenzhen

# Outline

- **HTML**
- **CSS**

# HTML

- Hyper Text Markup Language (HTML)
  - The language for publishing web pages on the World-Wide-Web
  - A Document Description Language , which is the standard markup language for documents designed due displayed in a web browser
  - Is a text document and is human-readable
  - Works together with CSS (Cascading Style Sheets) for layout and JavaScript for programming



A screenshot of a web browser window. The address bar shows the path '/Users/junhan/Downloads/example.html'. Below the address bar is a toolbar with various icons: a red 'Free' button, a purple circular icon, a green 'G' icon, a grey document icon, a blue arrow icon, a black puzzle piece icon, a black square icon, and a red circular icon with the Chinese character '俊' (Jun). The main content area displays a list of recent items: 'Apps', '从 IE 中导入', 'Visual Studio Cod...', 'AI Conference De...', 'Google C++ Style...', 'Inbox - jhan5@nd...', 'Google Scholar', 'https://console.cl...', 'CSRankings: Com...', 'Other Bookma...', and a 'More' button.

# My First Heading

My first paragraph.

# HTML document

the root element of  
an HTML page

```
<!DOCTYPE html>
```

declaration defines that this  
document is an HTML5 document

```
<html>
```

```
<head>
```

contains meta information  
about the HTML page

```
<title>Page Title</title>
```

specifies a title  
for the HTML page

```
</head>
```

```
<body>
```

defines the document's body

defines a  
large heading

```
<h1>My First Heading</h1>
```

```
<p>My first paragraph.</p>
```

defines a paragraph

```
</body>
```

```
</html>
```

# HTML page structure

```
<html>

    <head>

        <title>Page title</title>

    </head>

    <body>

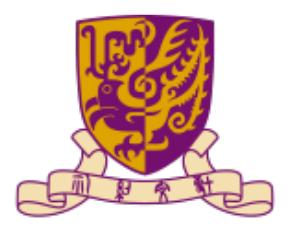
        <h1>This is a heading</h1>

        <p>This is a paragraph.</p>

        <p>This is another paragraph.</p>

    </body>

</html>
```



# HTML page structure

- <html lang="en">
  - Specify the language used to write the content of a web page
  - en is for English
  - zh is for Chinese
  - en-US, en-IN, en-GB
- <meta charset="utf-8">
  - Specify the character encoding for the HTML document



# HTML element

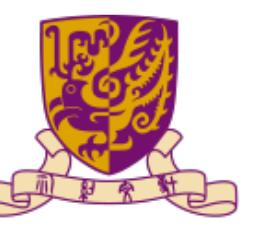
- An HTML document is a collection of elements having the form.  
`<tag> content </tag>`
- Elements can and frequently do nest  
`<tag1><tag2> content </tag2></tag1>`
- Self closing tags used when no content is necessary  
`<tag>`
  - E.g., `<img>` and `<br>`

# HTML element

- Defined by a start tag, some content, and an end tag

`<tagname> Content goes here... </tagname>`

Start tag	Element content	End tag
<code>&lt;h1&gt;</code>	My First Heading	<code>&lt;/h1&gt;</code>
<code>&lt;p&gt;</code>	My first paragraph.	<code>&lt;/p&gt;</code>
<code>&lt;br&gt;</code>	<i>none</i>	<i>none</i>



# HTML headings

- Defined with the `<h1>` to `<h6>` tags
- `<h1>` defines the most important heading and `<h6>` defines the least important headings

```
<!DOCTYPE html>
<html>
<body>

<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<h3>This is heading 3</h3>
<h4>This is heading 4</h4>
<h5>This is heading 5</h5>
<h6>This is heading 6</h6>

</body>
</html>
```

**This is heading 1**

**This is heading 2**

**This is heading 3**

**This is heading 4**

**This is heading 5**

**This is heading 6**

# HTML paragraphs

- Defined with the `<p>` tag

```
<!DOCTYPE html>
<html>
<body>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
</body>
</html>
```

This is a paragraph.

This is another paragraph.

# HTML links

- Defined with the `<a>` tag

```
<!DOCTYPE html>
<html>
<body>

<h2>HTML Links</h2>
<p>HTML links are defined with the a tag:</p>

<a href="https://www.w3schools.com">This is a link</a>

</body>
</html>
```

## HTML Links

HTML links are defined with the a tag:

[This is a link](#)

# HTML images

- Defined with the `<img>` tag

```
<!DOCTYPE html>
<html>
<body>

<h2>HTML Images</h2>
<p>HTML images are defined with the img tag:</p>

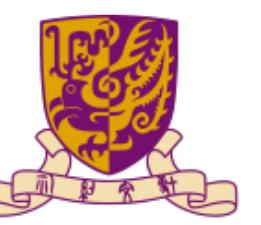


</body>
</html>
```

## HTML Images

HTML images are defined with the img tag:





# Debug on Web

The screenshot shows a Firefox browser window with the "Web Developer Tools" option selected in the "Browser Tools" menu. The browser interface includes a search bar, a toolbar with various icons, and a list of recent sites at the bottom. The developer tools panel is visible at the bottom of the screen.

Recent Sites:

- localhost
- i.cuhk
- hrms.cuhk
- sis.cuhk
- developer.m...
- sts.cuhk
- YouTube
- Facebook

Developer Tools Panel:

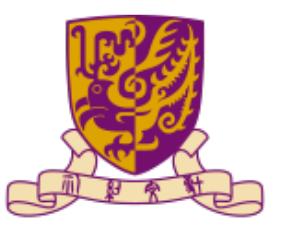
- Console (selected)
- Inspector
- Debugger
- Network
- Style Editor
- Performance
- Memory
- Storage
- Accessibility
- Application

Log Filter:

- Filter Output
- Errors
- Warnings
- Logs
- Info
- Debug
- CSS
- XHR
- Requests

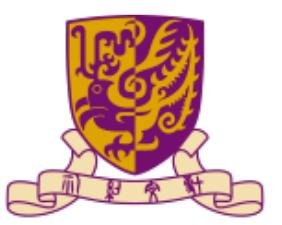
Page Number:

19



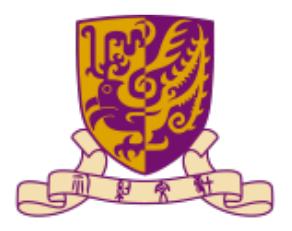
# HTML attributes

- href: a URL of a page link
- src: a path to the image
- width and height: image size
- style: color, font, size, and so on
- id: specify a unique name for an HTML element
- class: specify one or more class names for an HTML element
- etc.



# HTML id vs. class

- id
  - Suggest that each tag should have a unique id in an HTML document
  - Case sensitive
- class
  - Can be used by multiple HTML elements

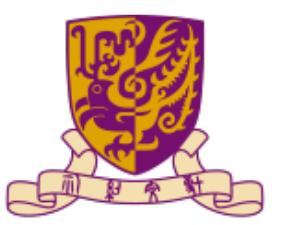


# HTML id

```
<h2 id="myheader">The id Attribute</h2>
<p>Use CSS to style an element with the id "myHeader":</p>
```

```
<h1 id="myHeader">My Header</h1>
<h2 id="myHeader">My Header 2</h2>
<script >
  h = document.getElementById("myHeader")
  console.log(h)
</script>
```

What is the output of h?



# HTML class

```
<h1 id="myHeader">My Cities</h1>

<h2 class="city">London</h2>
<p>London is the capital of England.</p>

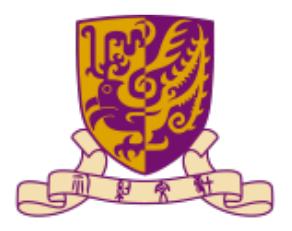
<h2 class="city">Paris</h2>
<p>Paris is the capital of France.</p>

<h2 class="city">Tokyo</h2>
<p>Tokyo is the capital of Japan.</p>

<script >
  h = document.getElementsByClassName("city")
  console.log(h)

</script>
```

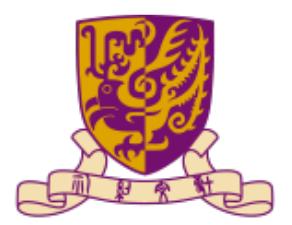
What is the output of h?



## HTML comment

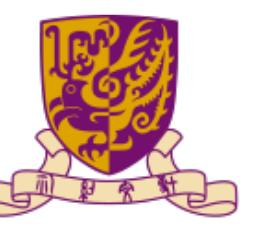
- HTML comments are enclosed between <!-- and -->

```
<!--  
<h1>hello, world</h1>  
-->
```



# HTML style

- text color
- background color
- fonts
- text size
- text alignment
- etc.



# HTML style

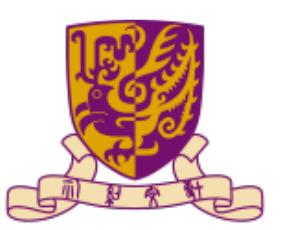
```
<!DOCTYPE html>
<html>
<body style="background-color:purple;">

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

## This is a heading

This is a paragraph.



# HTML style

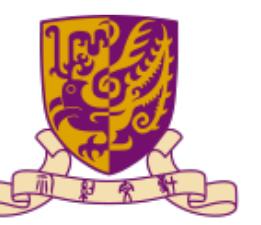
```
<!DOCTYPE html>
<html>
<body>

<h1 style="background-color:powderblue;">This is a heading</h1>
<p style="background-color:tomato;">This is a paragraph.</p>

</body>
</html>
```

# This is a heading

This is a paragraph.



# HTML style

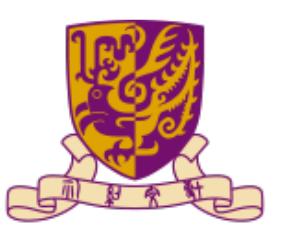
```
<!DOCTYPE html>
<html>
<body>

<h1 style="color:blue;">This is a heading</h1>
<p style="color:red;">This is a paragraph.</p>

</body>
</html>
```

## This is a heading

This is a paragraph.



# HTML style

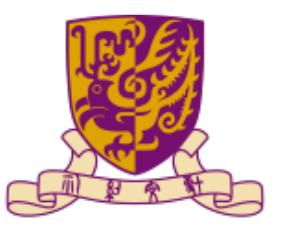
```
<!DOCTYPE html>
<html>
<body>

<h1 style="text-align:center;">Centered Heading</h1>
<p style="text-align:left;">Centered paragraph.</p>

</body>
</html>
```

## Centered Heading

Centered paragraph.

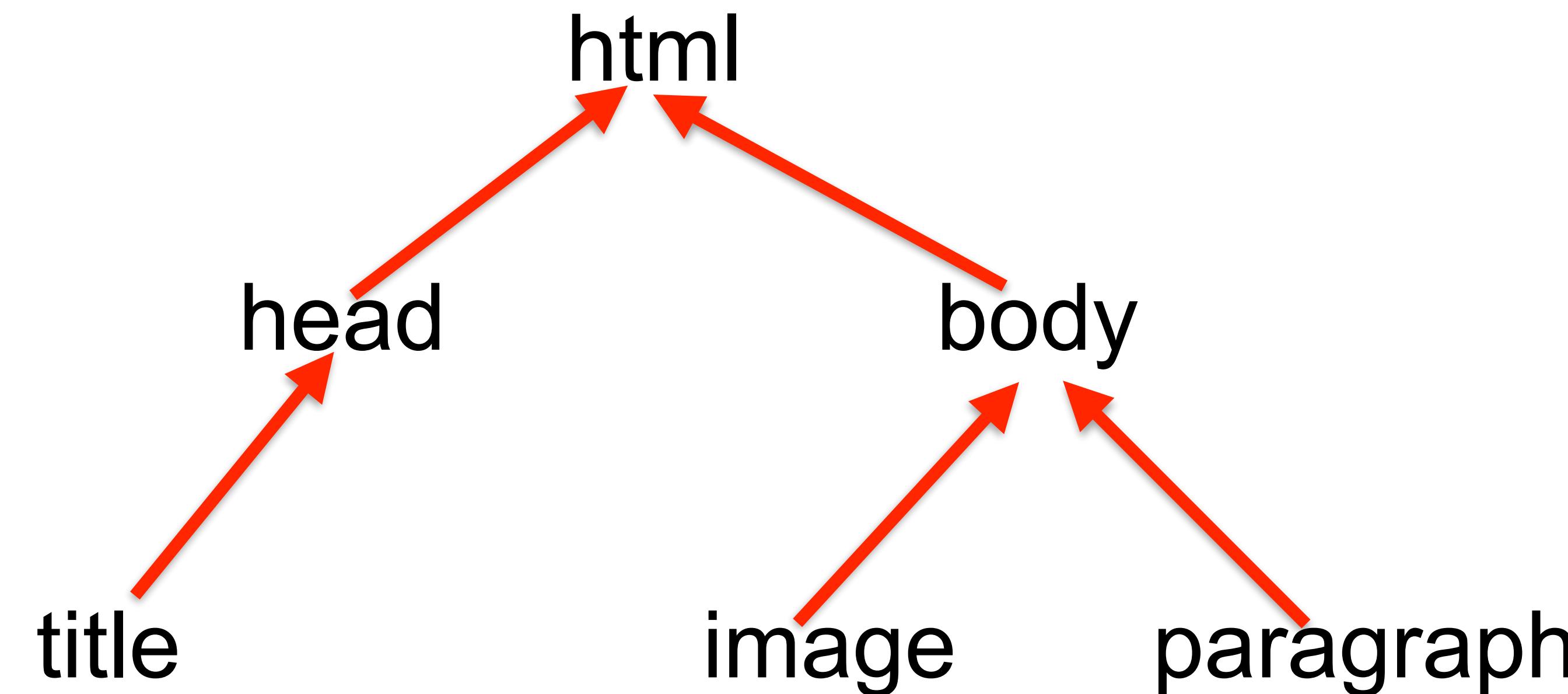


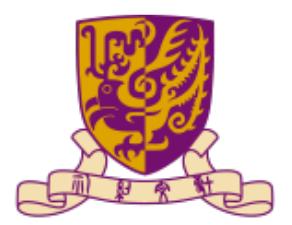
# Good HTML syntax

- There are lots of ways to encode HTML that your browser frequently will be able to handle
- A syntax checker can often be helpful in debugging: <http://validator.w3.org/>

# DOM

- Stand for Document Object Model
- A programming interface for web document
- The nested structure of tags form a tree, with the root element being `<html></html>` and then all of its children contained within





## HTML exercises

- Stylize “`<h1>Centered Heading</h1>`” so that the text is aligned in center
- Stylize `<p>This is a paragraph.</p>` to change text font
- Use `<ul>` and `<li>` to generate a list



# HTML exercises

- <h1 style="text-align:center;">Centered Heading</h1>
- <p style="font-family:courier;">This is a paragraph.</p>

```
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```



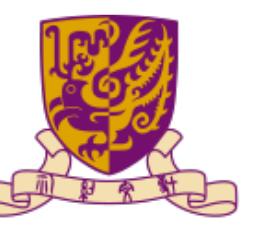
香港中文大學(深圳)

The Chinese University of Hong Kong, Shenzhen

# Outline

- HTML
- CSS

- Cascading Style Sheets
  - Control the layout of multiple web pages at once



```
<!DOCTYPE html>
<html>
<body>

<h1 style="color:blue;">This is a heading</h1>
<p style="color:red;">This is a paragraph.</p>

</body>
</html>
```

# This is a heading

This is a paragraph.

# Content vs. appearance

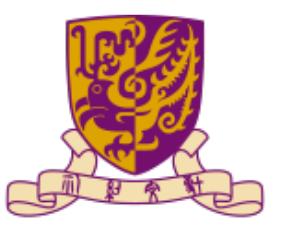
- The HTML document provides a decomposition of the content of the document, but with few exception does not indicate how a browser should display it
- CSS provide mechanisms to control appearance (style) in a very fine granularity
- CSS is rule-based, using names, ids, and classes to specify rules that applied sequentially

# CSS rules

- General format

```
selector {  
    property1: value;  
    property2: value;  
}
```

- Selectors can be
  - Names (to apply to all tags of that name) name {...}
  - IDs (applying to specific tags with that unique id) #id {...}
  - Classes (to allow for user-defined groups) class {...}



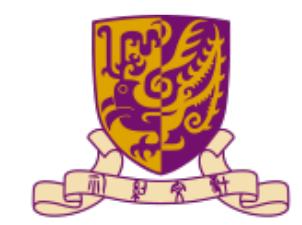
# Inline styles

```
<!DOCTYPE html>
<html>
<body>
  <p style="font-size:66px; font-family:monospace">This paragraph uses 66px
monospace font.</p>
  <p>This paragraph uses default font.</p>
  <p>This paragraph uses <span style="font-size:53.3px">53.3px inside this span</
span>
    but default font size here.</p>
</body>
</html>
```

This paragraph uses 66px monospace font.

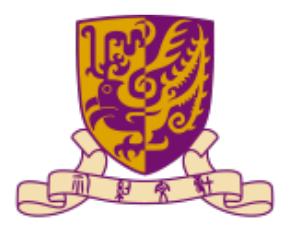
This paragraph uses default font.

This paragraph uses 53.3px inside this span but default font size here.



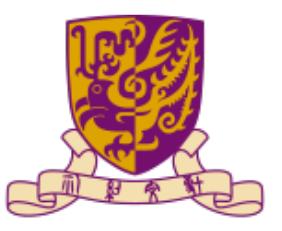
# Internal CSS

```
<!DOCTYPE html>
<html>
<head>
<style>
  body {
    background-color:cyan;
  }
  h2 {
    color:white;
    background-color:black;
  }
  p.monospace {
    font-size:16px;
    font-family:monospace;
  }
  p.f20px {
    font-size:20px;
  }
</style>
</head>
<body>
  <h2>H2 is white on black</h2>
  <p>This paragraph is normal.</p>
  <p class="monospace">This paragraph uses 16-px monospace font.</p>
  <p class="f20px">This paragraph uses 20-px font.</p>
</body>
</html>
```



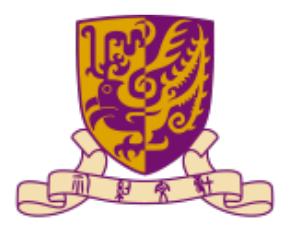
# External CSS

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <link href="TestExternal.css" rel="stylesheet">
</head>
<body>
  <h2>H2 is white on black</h2>
  <h2 id="green">This H2 is green on black</h2>
  <p>The default paragraph uses 12-pt small-cap font.</p>
  <p class="f24pt">This paragraph uses 24-pt, italics font with text-indent of 1cm.
  It inherits the small-cap property from the default paragraph selector.</p>
</body>
</html>
```



# CSS style conflict

- Style conflict on an element arises
  - A property is inherited from multiple ancestors
  - More than one rules are applicable to the element. For example, Tag-selector p, Class-selector .red and Id-selector #comment are all applicable to element <p id="comment" class="red">.



# CSS style conflict

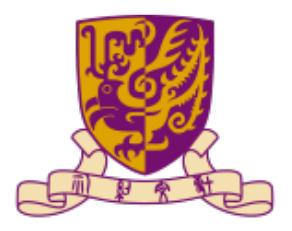
- Nearest ancestor wins
  - If a property is not defined for an element and is inheritable, it will be inherited from the nearest ancestor

```
<head>
  <style>
    p { color:black; background-color:lightblue; }
    .red { color:red;  }
    #comment { color:yellow;}
  </style>
</head>
<body>
  <p id="comment" class="red"> This is a test.</p>
</body>
```

# CSS style conflict

- Nearest ancestor wins
- Specificity
  - Specificity means that "the more specific the selector, the stronger the rule"

```
<head>
<style>
p { color:black; background-color:white; }
/* Override the color properties */
p.red { color:red; }
p#id1 { color:yellow; background-color:lightblue; }
p#id2 { color:blue; }
p#id1 { color:green; }
</style>
</head>
<body>
<p id="id1">Paragraph with id of "id1" (green)</p>
<p id="id2">Paragraph with id of "id2" (blue)</p>
<p class="red">Paragraph of class of "red" (red)</p>
<p id="id1" class="red">Paragraph with id of "id1" and class of "red" (green)</p>
<p>Paragraph without id and class with default colors (black)</p>
</body>
```



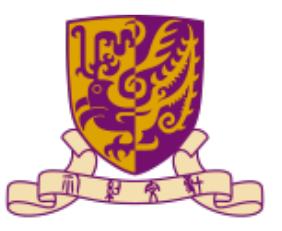
# CSS style conflict

- Nearest ancestor wins
- Specificity
- Locality
  - If the "Law of Specificity" cannot resolve the conflict, apply the "Law of Locality". The style-rule that read in last by the browser takes effect. In the above example, there are two ID-selector for id1, the latter takes effect.
  - The inline style (applied to a specific tag via style attribute) overrides the internal style (defined in <style>) and external style sheet (defined via <link>). For internal and external styles, the order of <link> and <style> elements determine the precedence. It is recommended to place the <link> before <style> so that the internal styles can override the external styles.

# HTML attributes id and class

- All the HTML elements supports two optional attributes: id="id-value" and class="class-value"
- You can assign an id="id-value" attribute to an HTML element to uniquely identify that element. The id-value must be unique within the HTML document. In other words, no two elements can have the same id-value. The id attribute can be used by CSS (as well as JavaScript) to select that particular element.

```
<div id="header"><h1>Header Section</h1> ..... </div>
<div id="content"><h1>Content Section</h1> ..... </div>
<div id="footer"><h1>Footer Section</h1> ..... </div>
```

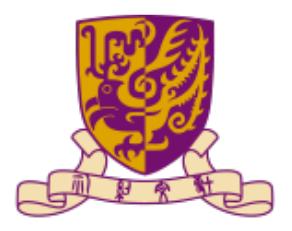


# HTML attributes id and class

- All the HTML elements supports two optional attributes: `id="id-value"` and `class="class-value"`
- Similarly, you can assign `class="class-value"` attribute to a class of elements having the same presentation properties and appearance. The class-value needs not be unique. That is, the same class-value can be assigned to many HTML elements. In other words, these HTML elements form a sub-class (hence, the keyword `class`). The `class` attribute is primarily used by CSS to apply a common set of styles to all the elements of the same class.

# HTML id vs class attributes

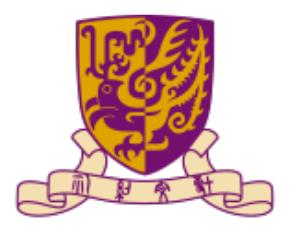
- id and class are used for reference HTML elements by Javascript and CSS
- id must be unique in an HTML document, but class needs not
- An element can have one id, but many classes
- CSS's Id-selector begins with # (e.g., #header), Class-selector begins with dot (.) (e.g., .header)



# CSS tag-selector

- A CSS tag-selector selects HTML elements based on the tag-name

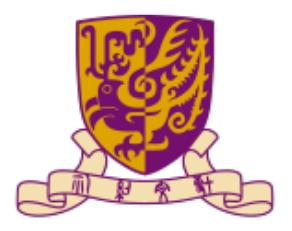
```
/* tag selector */  
h2 { background-color:black; color:white; text-align:center; }
```



# CSS id-selector

- A CSS id-selector, which begins with a '#' followed by an id value, selects a unique element (because id value is supposed to be unique) in the document

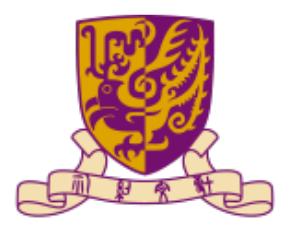
```
/* ID-selector (id-value is unique in a document) */  
#header { font-size:18px; color:cyan; }  
#content { font-size:14px; color:black; }  
#footer { font-size:12px; color:orange; }
```



# CSS class-selector

- A CSS class-selector, which begins with a '.' followed by a classname, selects ALL elements having that class value.

```
/* Class-selector (class value needs not be unique) */  
.highlight { background-color: #ff0; }  
.underline { text-decoration: underline; }  
.green    { color:green; text-decoration:underline; }  
.blue     { color:blue; }
```



# CSS group selector

- You can apply the same style definitions to multiple selectors, by separating the selectors with a commas ','

```
h1, h2, h3 { background-color:black; color:white; }
```

# CSS descendant selector

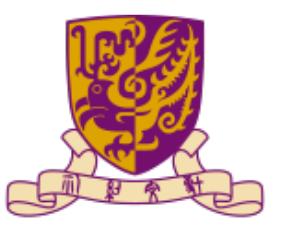
- You can define a style rule that takes effect only when a tag occurs within a certain contextual structure, e.g., descendant, immediate-child, first-child, sibling, etc
- To create a descendant selector, list the tags in their hierarchical order, with no commas separating them (commas are meant for grouping selectors).

```
ul li { color:red; }  
ul ul li { color:blue; }  
ul ul ul li { color:green; }
```

# CSS tag-cum-class selector

- The selector T.C selects all tag-name T with classname of C. This is a restricted form of the class selector, which applies to the specific tag-name only.

```
p      { color:black; } /* default style for all <p> tags */  
p.red  { color:red; } /* applicable to <p class="red"> tags (override default) */  
p.blue { color:blue; } /* applicable to <p class="blue"> tags (override default) */  
h1, h2, h3 { color:green; } /* default style for <h1>, <h2> and <h3> tags */  
h3.white { color:white; } /* applicable to <h3 class="white"> tags (override default) */  
h3.upper { text-transform:uppercase; }  
<p>This paragraph is in black (default style)</p>  
<p class="red">This paragraph, of class="red", is in red.</p>  
<p class="blue">This paragraph, of class="blue", is in blue.</p>  
<h2>H2 in green (default style)</h2>  
<h3>H3 in green (default style)</h3>  
<h3 class="white upper">This H3, of class="white", is in white</h3>
```



# CSS universal selector

- The universal selector \* selects ALL the elements in the document.

```
* { margin:0; padding:0; } /* all tags have margin and padding of 0 */
```

# CSS pseudo-class selector

- Pseudo-class describes a special state for an element, e.g. hover and focus. CSS defines a number of pseudo-classes for anchor elements `<a>`, namely, `a:link` (unvisited link), `a:visited` (visited link), `a:focus` (on focus), `a:hover` (mouse pointer hovers over), `a:active` (clicked or active link). Take note that colon ":" is used to identify pseudo classes instead of "." for ordinary classname
- These pseudo classes is commonly-used with the `<a>` element. But `:hover`, `:focus`, and `:active` can also be applied to other elements, such as `<p>`, `<li>`, and etc.

# CSS pseudo-class selector

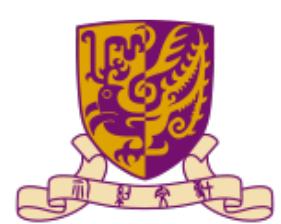
- The order is important for anchor pseudo-classes in applying styles. It shall be link-visited-focus-hover-active (LVFHA)
- It is called pseudo-class, as it sub-divide the <a> tag into four sub-classes that cannot be marked out manually
- You can further restrict the selections via a.classname:pseudo-classname

```
a      { font-size:14px; } /* all <a> tags */  
a:link { color:red;    } /* unvisited link */  
a:visited { color:green; } /* visited link */  
a:focus { color:lightblue; } /* on focus via tab key */  
a:hover { color:blue;   } /* mouse over link */  
a:active { color:black; } /* currently selected link */
```

# CSS pseudo-elements selector

- Pseudo-class describe a special state, e.g., hover or focus.  
Pseudo-Element is a part of an element, e.g., first-letter, first-line (of a <p>). Pseudo-class selector begins with colon (:), whereas pesudo-element selector begins with double colons (::) in CSS3 to differentiate with the pesudo-class selector

```
.title::before {  
    content: "\0022"; /* Add a double quote before */  
    color: "red";  
}  
.title::after {  
    content: "\0022"; /* Add a double quote after */  
    color: "red";  
}
```



# Types of CSS selectors

Selector	Examples	Description
*	* { }	Universal Selector: select ALL elements in the document
tag-name	h1 { }	Tag Selector: select all <h1> elements
#id-name	#header { }	ID Selector: select the unique element with id="header"
.class-name	.new { }	Class Selector: select elements with class="new"
:pseudo-class	:first-child { } :focus { } a:active { }	Pseudo-Class Selector: select a special state such as focus and hover.
::pseudo-element	::first-line { } ::first-letter { } ::before { } ::after { } ::selection { }	Pseudo-Element Selector: select a part of an element and may modify the selected element.



# Types of CSS selectors

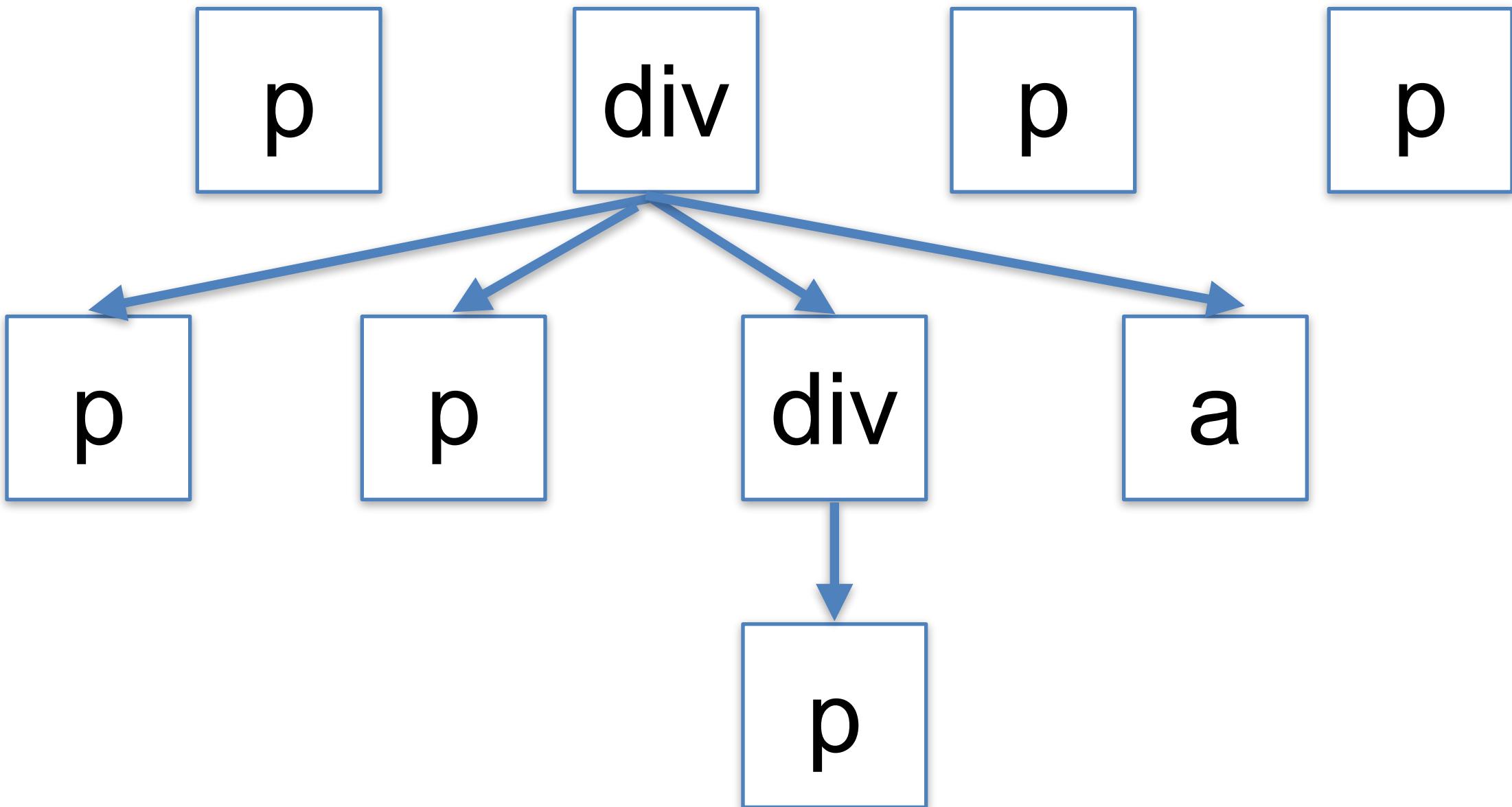
Selector	Examples	Description
S1, S2, S3	h1, h2 { }	Group Selector: Apply the style to S1, S2 and S3
S1 S2	div p { }	Descendant Selector (separated by a space): select if S2 is a descendant (child, grandchild, etc.) of S1
S1>S2	tr > td { }	Child (Direct Descendant) Selector: only if S1 is an immediate child of S1
S1:S2	ul:li { }	First-Child Selector: only if S2 is the first child of S1
S1+S2	a + span { }	Adjacent Sibling Selector: the (one) S2 that is immediately after S1
S1~S2	a ~ span { }	General Sibling Selector: All S2(s) that are siblings after S1
T.C	div.example { }	Tag cum Class Selector: <div class="example">
T#C	div#header { }	Tag cum ID Selector: <div id="header">
.C1.C2	.new.example { }	Multiple Class Selector (no space in between): <div class="C1 C2"></div>

# CSS selector

- **#section1 p**

```
<p> Section 1 </p>
<div id = 'section1'>
  <p> 1.1 </p>
  <p> 1.2 </p>
  <div>
    <p> 1.2.1 </p>
  </div>
  <a href = "#"> 1.3 </a>
</div>
<p> Section 2 <p>
<p> Section 3 <p>
```

Descendant Selector (separated by a space):  
select if S2 is a descendant (child, grandchild, etc.) of S1

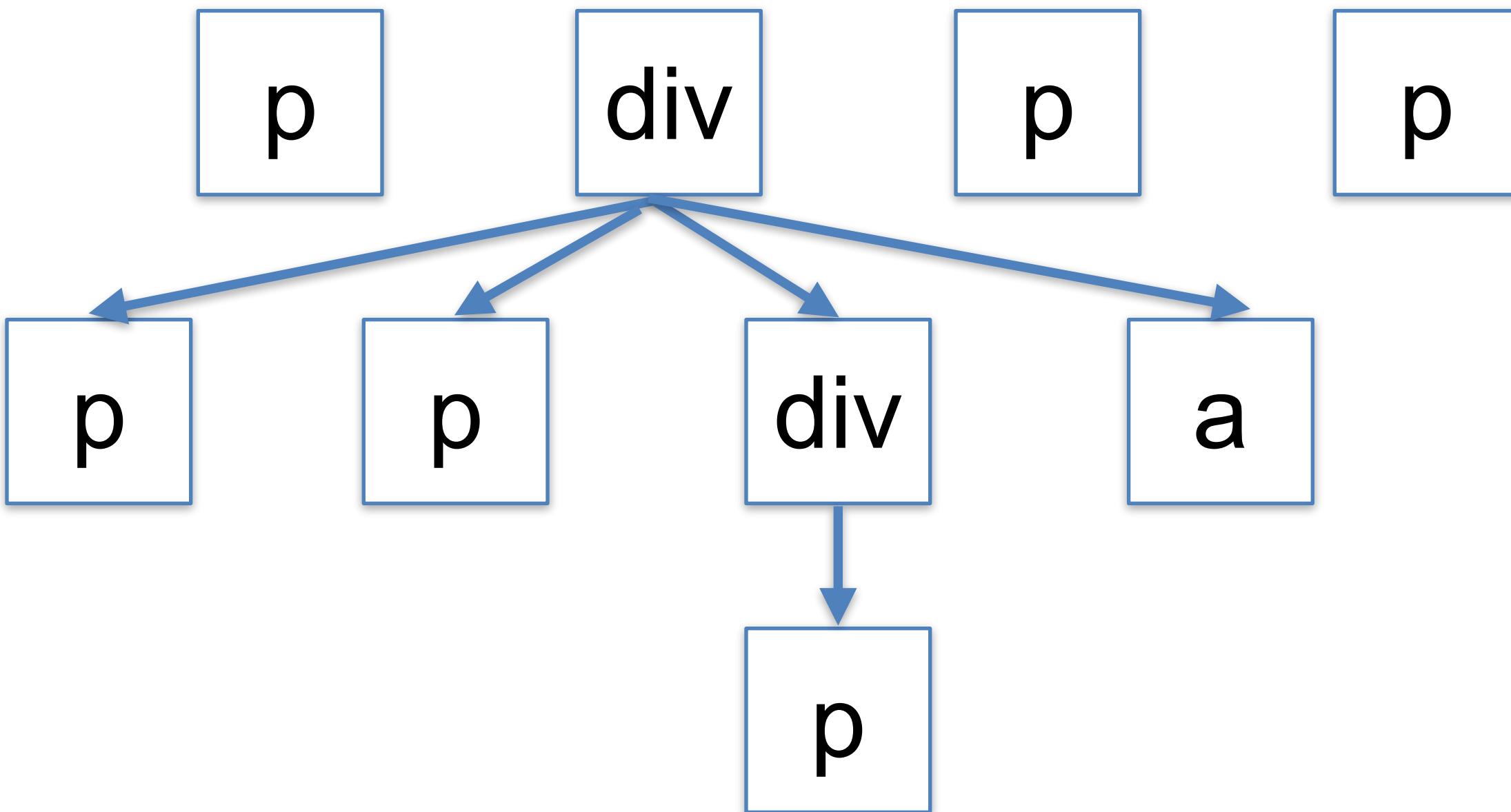


# CSS selector

- **#section1>p**

Child (Direct Descendant) Selector: only if S1 is an immediate child of S1

```
<p> Section 1 </p>
<div id = 'section1'>
  <p> 1.1 </p>
  <p> 1.2 </p>
  <div>
    <p> 1.2.1 </p>
  </div>
  <a href = "#"> 1.3 </a>
</div>
<p> Section 2 <p>
<p> Section 3 <p>
```

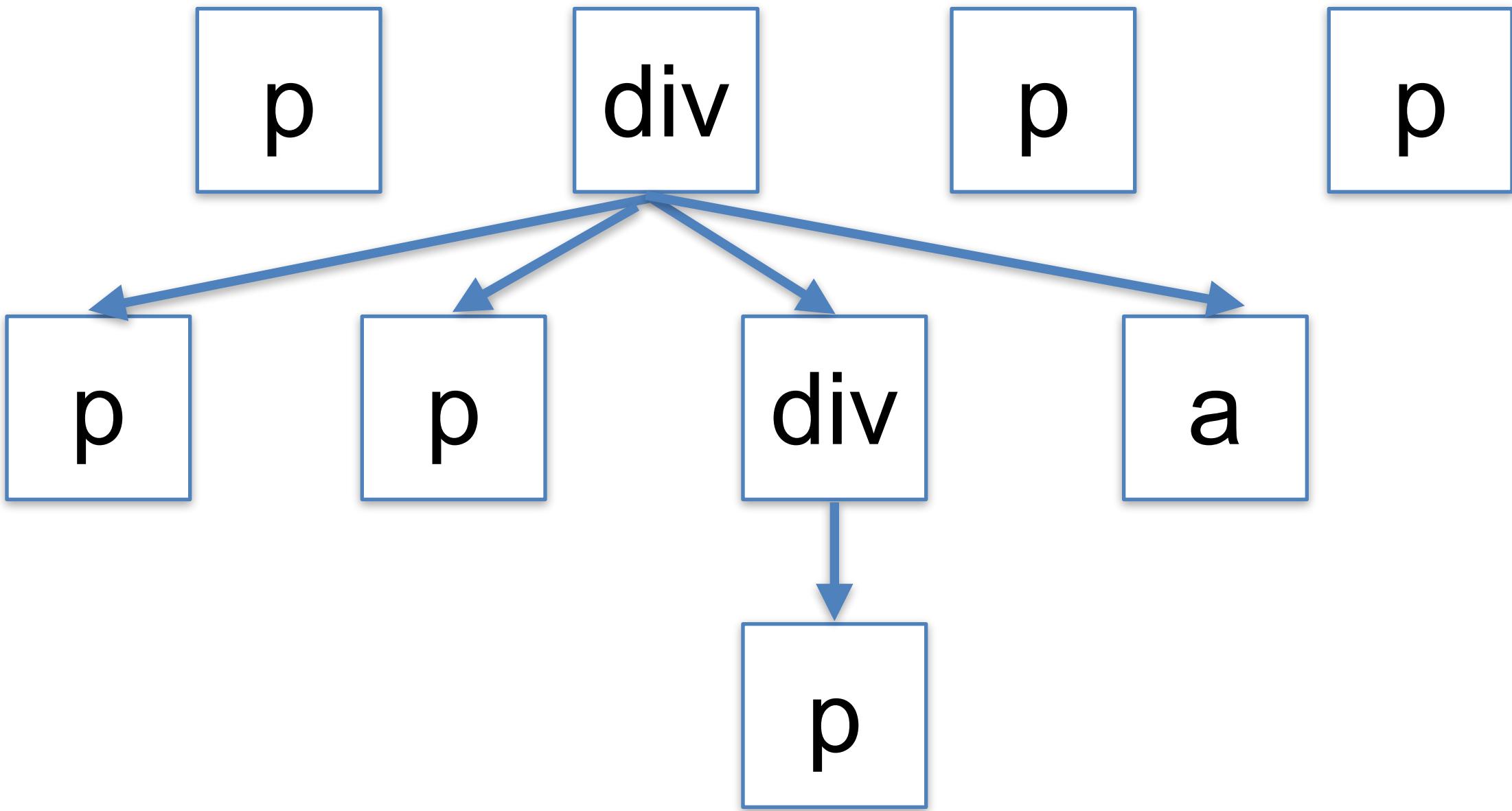


# CSS selector

- p:first-child

```
<p> Section 1 </p>
<div id = 'section1'>
  <p> 1.1 </p>
  <p> 1.2 </p>
  <div>
    <p> 1.2.1 </p>
  </div>
  <a href = "#"> 1.3 </a>
</div>
<p> Section 2 <p>
<p> Section 3 <p>
```

First-Child Selector: only if S2 is the first child of S1

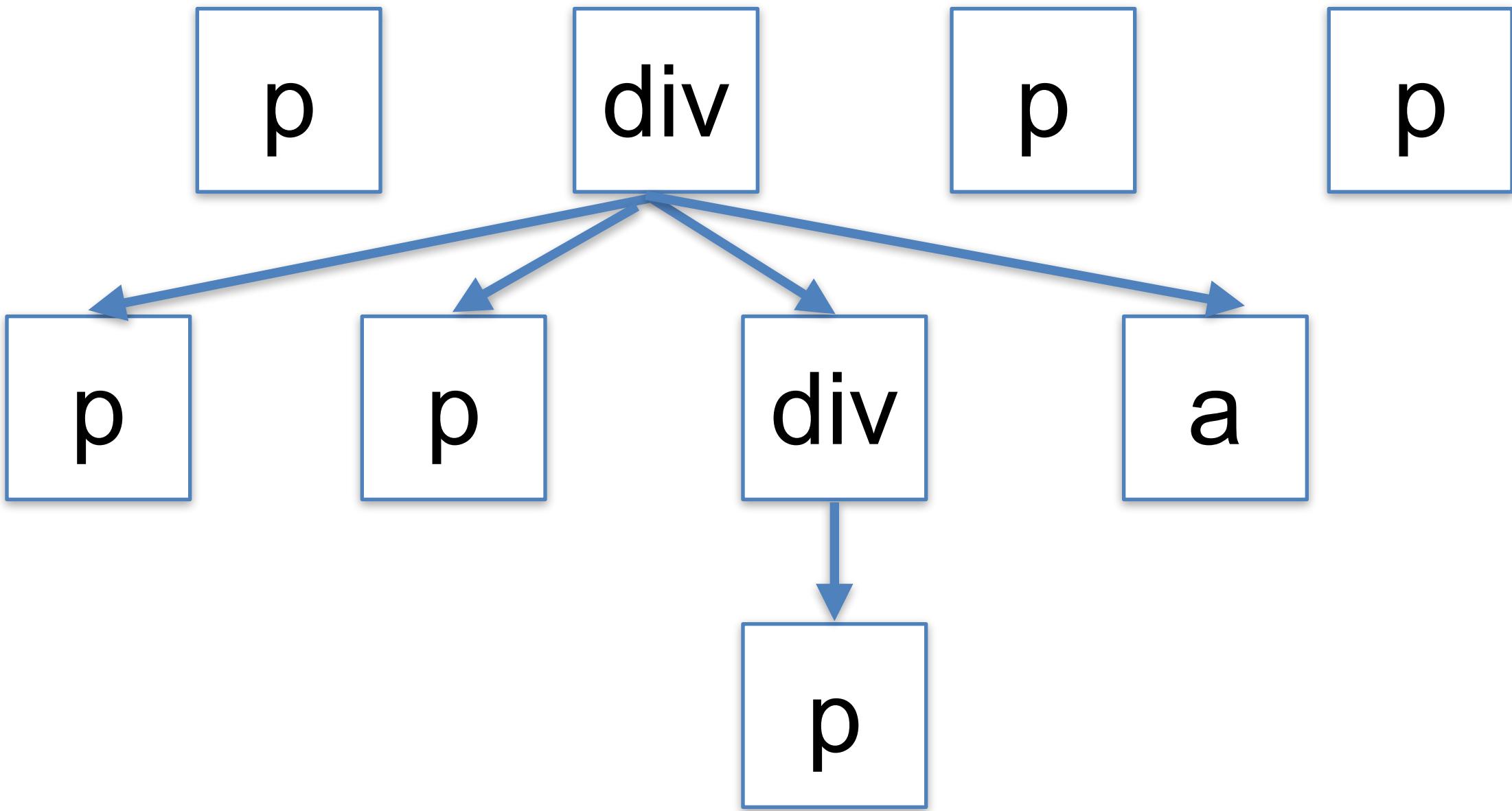


# CSS selector

- **#section1+p**

Adjacent Sibling Selector: the (one) S2 that is immediately after S1

```
<p> Section 1 </p>
<div id = 'section1'>
  <p> 1.1 </p>
  <p> 1.2 </p>
  <div>
    <p> 1.2.1 </p>
  </div>
  <a href = "#"> 1.3 </a>
</div>
<p> Section 2 <p>
<p> Section 3 <p>
```

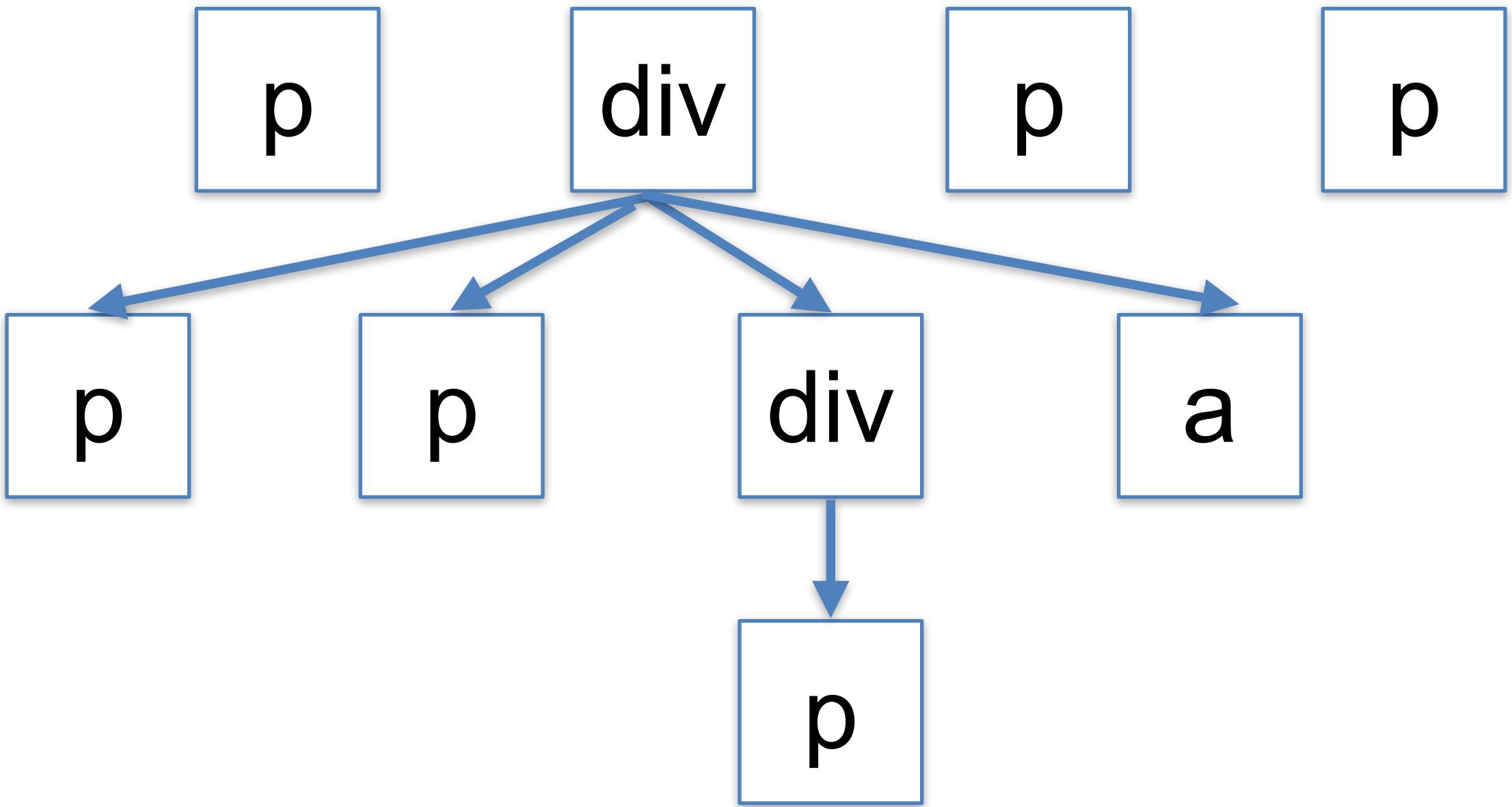


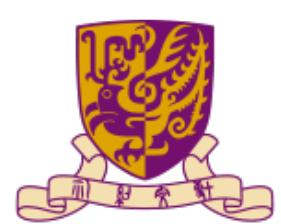
# CSS selector

- `#section1~p`

General Sibling Selector: All S2(s) that are siblings after S1

```
<p> Section 1 </p>
<div id = 'section1'>
  <p> 1.1 </p>
  <p> 1.2 </p>
  <div>
    <p> 1.2.1 </p>
  </div>
  <a href = "#"> 1.3 </a>
</div>
<p> Section 2 <p>
<p> Section 3 <p>
```



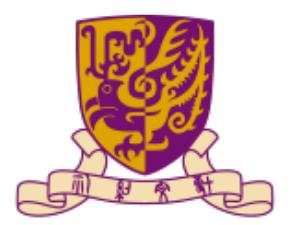


# Types of CSS attribute selectors

Selector	Example	Description
T[att]		Select tags with the attribute, regardless of its value
T[att='value']		Select tags with the attribute and value
T[att^='value']		Select tags with the attribute, beginning with value
T[att\$='value']		Select tags with the attribute, ending with value
T[att*= 'value']		Select tags with attribute, containing the value
T[att~= 'value']		Select tags with attribute having value in the list space-separated values.
T[att = 'value']		Select tags with attribute, beginning with value followed by dash (-)
T[att='value' i]		Select tags with attribute and value, where value is case-insensitive

# CSS inheritance

- Many (but not all) CSS properties, such as color and font-family, affect not only the elements selected by the selector, but also inherited by their descendants
- Inheritance is a big time-saver for designing styles
- Some properties (such as border, margin, padding, width, height, background-color) are not inherited



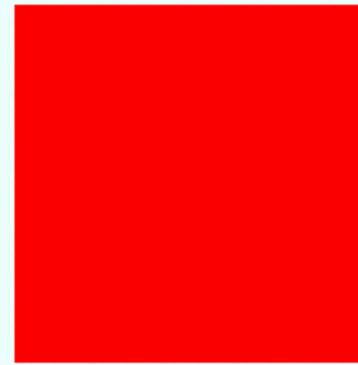
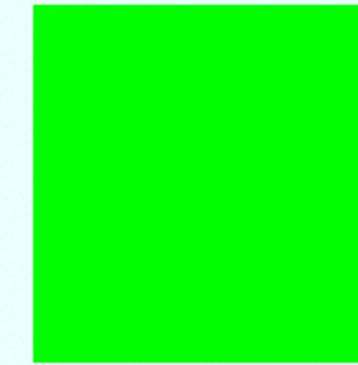
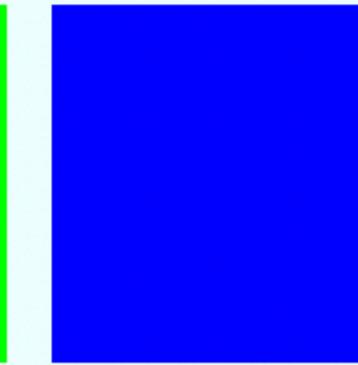
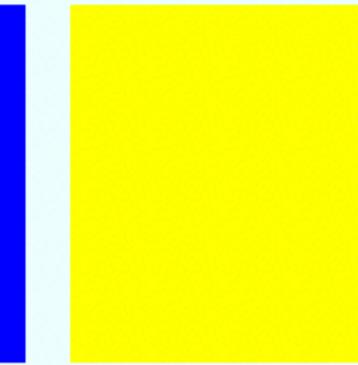
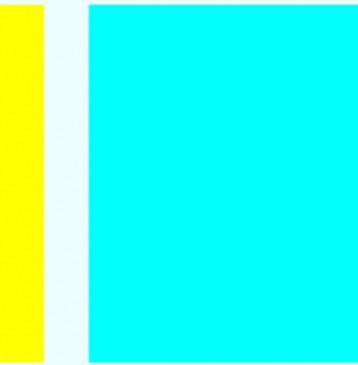
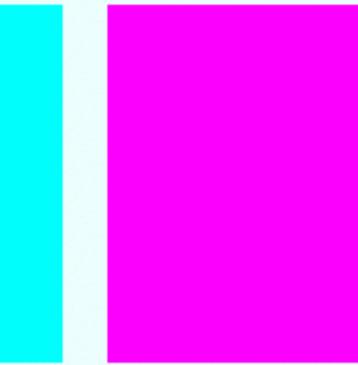
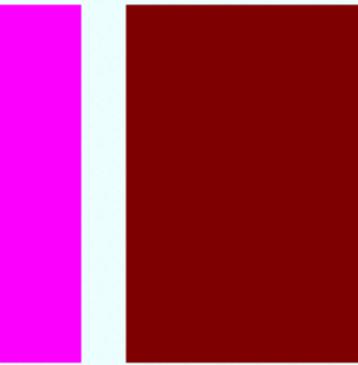
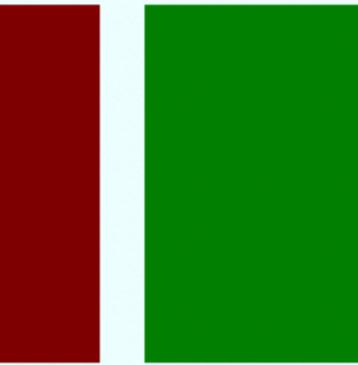
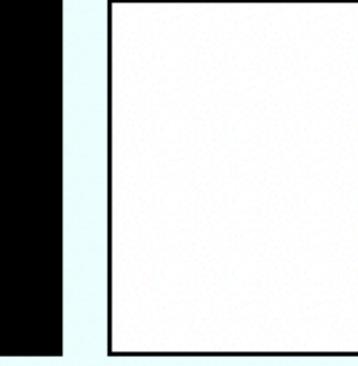
# CSS inheritance

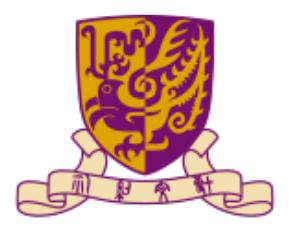
```
span {  
    color: blue;  
    border: 1px solid black;  
}  
  
.extra span {  
    color: inherit;  
}  
</style>  
</head>  
<body>  
  
<div>  
    Here is <span>a span element</span> which is blue, as span elements are set to be.  
</div>  
  
<div class="extra" style="color:green">  
    Here is <span>a span element</span> which is green, because it inherits from its parent.  
</div>  
  
<div style="color:red">  
    Here is <span>a span element</span> which is blue, as span elements are set to be.  
</div>
```

# CSS color and background-color properties

- Color can be expressed as
  - RGB in the form of `rgb(r, g, b)`. The r, g, b can be expressed in a decimal value between 0 and 255; or in percentage between 0% and 100%
  - RGBA in the form of `rgba(r, g, b, a)`: RGB with an additional A (alpha channel). The A is used to control the transparency-opacity, with  $a=1$  for opaque; and  $a=0$  for totally transparent
  - HSL in the form `hsl(hue, saturation, lightness)`: Hue is the color on the color wheel in degrees between 0 to 360. Saturation (purity of color) is expressed in percentage between 0% and 100% (pure color). Lightness (brightness or intensity) is also expressed in percentage between 0% (darkest) and 100% (brightest)
  - HSLA in the form of `hsla(hue, saturation, lightness, alpha)`

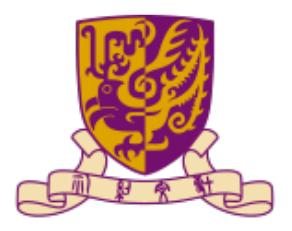
# CSS color and background-color properties

red #FF0000	lime #00FF00	blue #0000FF	yellow #FFFF00	aqua #00FFFF	fuchsia #FF00FF	maroon #800000	green #008000
							
navy #000080	olive #808000	teal #008080	purple #800080	black #000000	white #FFFFFF	gray #808080	silver #C0C0C0
							



# CSS color properties

- The most important color properties are color and background-color:
  - color: #rrggbb|rgb(r,g,b)|rgba(r,g,b,a)|color-name: Set the color of the text (or foreground). Color values are inherited by descendants.
  - background-color: #rrggbb|rgb(rrr,ggg,bbb)|rgba(r,g,b,a)|color-name|transparent|inherit|initial: Set the background color of an element. The default is transparent and NOT inherited, so as to create a see-through effect. The initial sets to its default value.



# CSS length measurements

- Many CSS properties, such as width, height, margin, border, padding, font-size and line-height, require a length measurement

```
html {  
    font-size: 16px; /* base measurement for rem (CSS3) */  
}  
p {  
    font-size: 1rem; /* relative to root html element */  
    width: 80%; /* 80% of the parent's width */  
    margin: 0.5em 2em; /* relative to current font-size */  
    border: 5mm; /* absolute millimeters */  
    padding: 0;  
    line-height: 140%; /* 1.4 times of the current font-size */  
}
```

# CSS length measurements

- Two types of length measurements: relative (to another length property) and absolute (e.g., inches, centimeters, millimeters)
- The absolute units are:
  - in (inch)
  - cm (centimeter)
  - mm (millimeter)
  - pt (point): 1 inch has 72 points.  $1\text{pt} \approx 0.014\text{in} \approx 0.35\text{mm}$
  - pc (pica): 1 pica is 12 points. 1 inch has 6 picas.  $1\text{pc} \approx 0.17\text{in} \approx 4.2\text{mm}$ .  
pc is not commonly used
  - px: px is a measurement unit created for the CSS, where the thinnest line shall have width of 1px. Today, it is defined as  $1\text{px}=1/96\text{ inch}$ . Since  $1\text{pt} \approx 0.014\text{in}$ ;  $12\text{pt} = 16\text{px} = 1/6\text{ inch} = 0.42\text{cm}$

# CSS length measurements

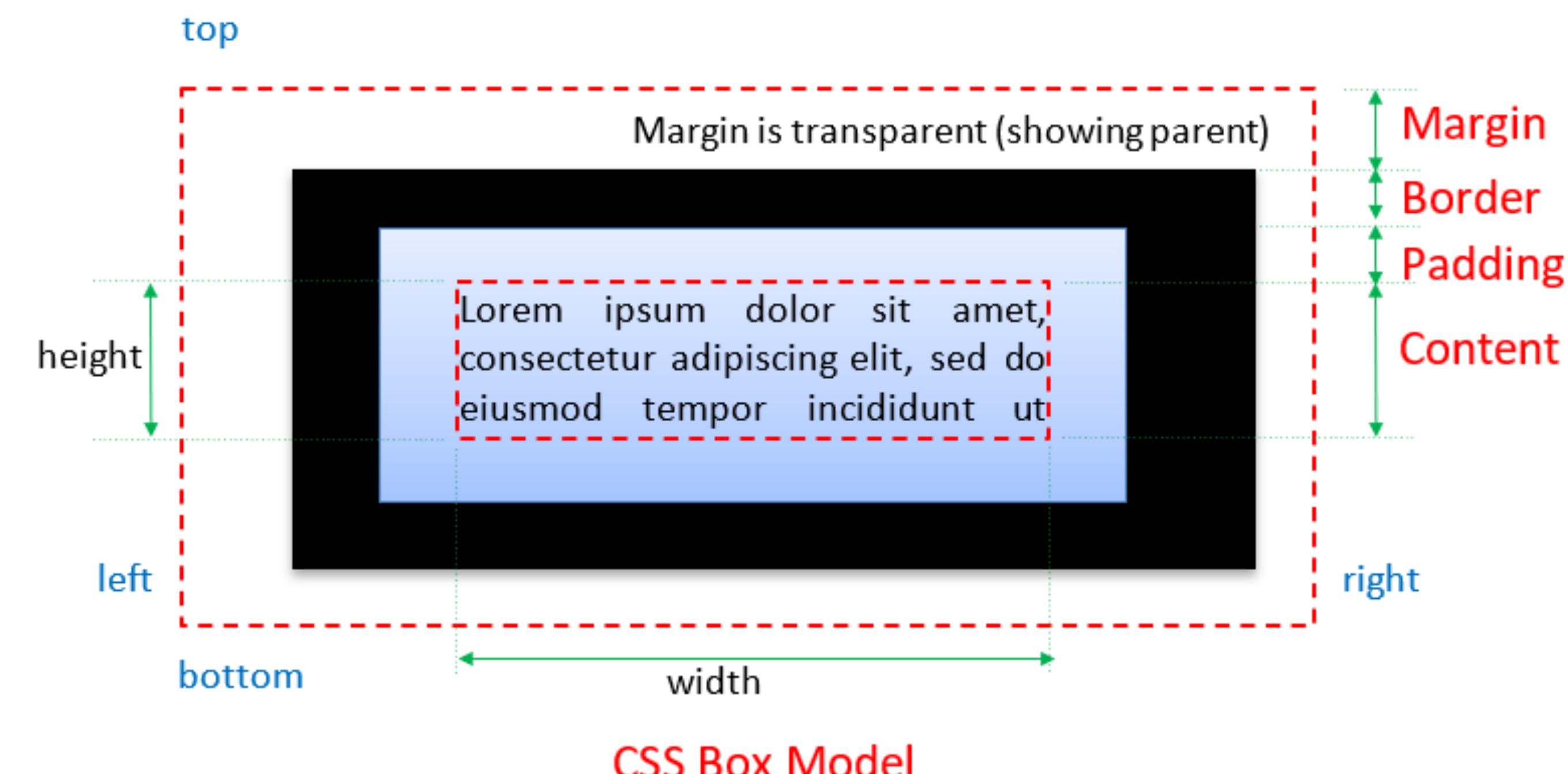
- The relative units are:
  - % (percent): in term of the percentage of a property of a referenced element, generally, the same property of the parent element
  - For example, table {width:80%} set the table's width to 80% of the width of the parent (probably a <div> or <body>)
  - em: the width of the letter 'm' of a referenced font, generally, the current font
  - For example, margin:2em means that the margins are twice the current (referenced) font-size.
  - However, if em is used to set the font-size property, it needs to find a reference. In this case, it is referenced to the parent's font-size. For example, p {font-size:1.2em} sets the font-size of <p> to 1.2 times of the parent (possibly a <div> or <body>)
  - rem (CSS3): relative to the font-size of the root or <html> element
  - vw, vh (CSS3): one percent of viewport width and height respectively
  - vmin, vmax (CSS3): one percent of the viewport smaller dimension or larger dimension respectively, i.e., 1vmin is equal to the smaller of 1vh or 1vw; 1vmin is the larger of 1vh or 1vw.
  - ex (not commonly-used): the height of letter 'x' of the parent's font. ex is not commonly used

# CSS length measurements

```
h6 {  
    font-size: 1.2rem; /* 1.2 times of the <html> font-size */  
    width: 80%; /* 80% of the parent's width */  
    margin: 0.5em 1.2em; /* relative to the current font's letter 'm' */  
    padding: 10px; /* 10 logical pixels */  
    border: 0; /* zero does not need a unit */  
}  
  
line-height: 20px; /* 20 pixels */  
line-height: 150%; /* 150% of the parent's line-height */  
line-height: 1.2em; /* 1.2 times of the current font's letter 'm' */  
line-height: 1.5; /* 1.5 times of the current font */
```

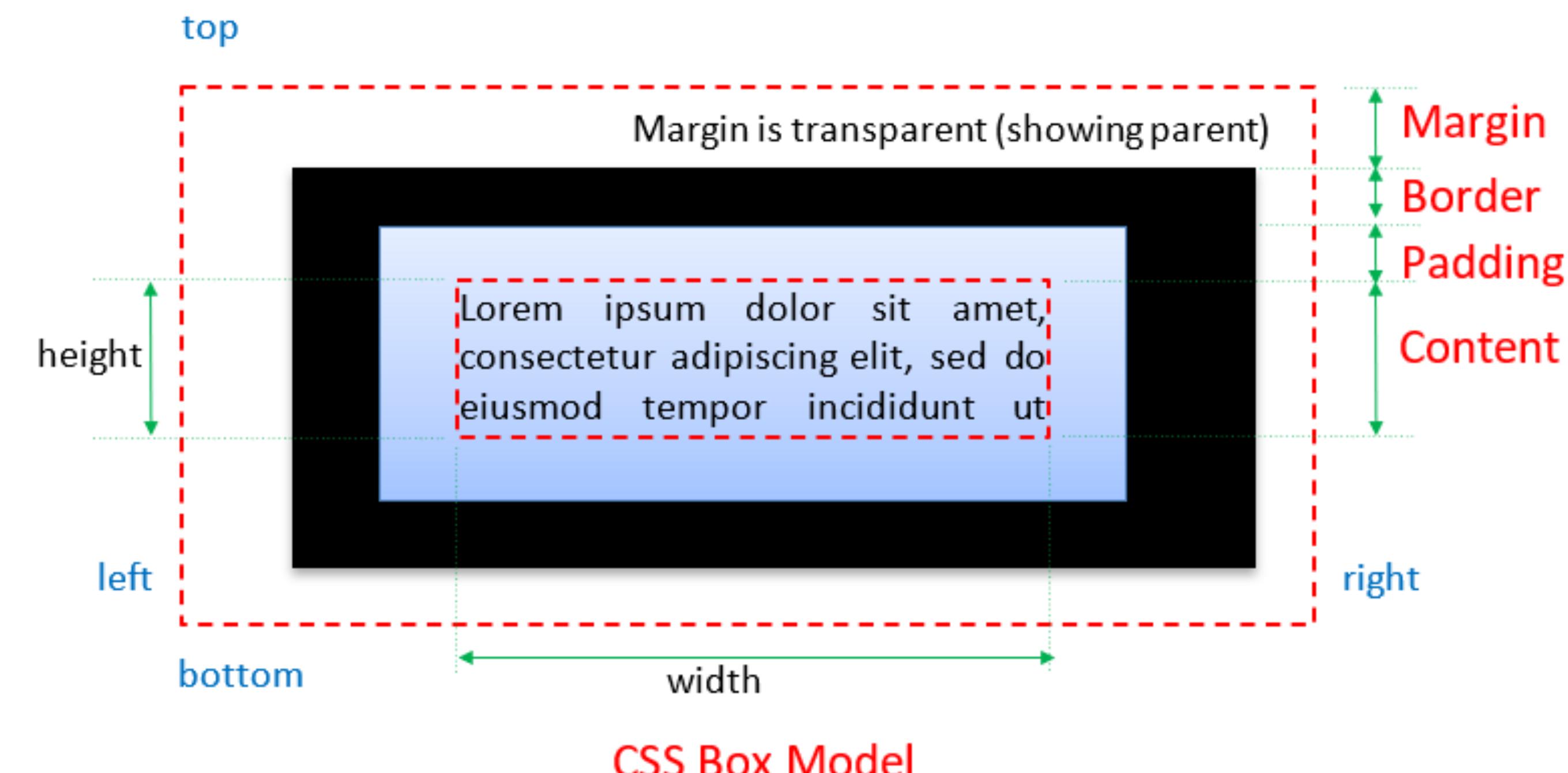
# CSS box model

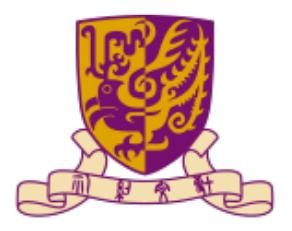
- The content area contains the texts, image, or child elements.
- The padding is the space between the content area and the border. It clears an area outside the content area. It has the same background as the content area.



# CSS box model

- The border goes between padding and margin. You can set a color and a style (such as solid, dash, dotted) to the border.
- The margin is the space outside the border (to another element). It clears an area outside the border. The margin does not have a background, and is totally transparent.





# CSS box model

```
#elm {  
    width: 300px;  
    margin: 10px;  
    border: 5px solid black;  
    padding: 20px;  
}
```

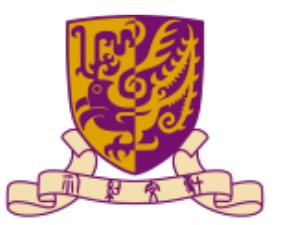
The actual width of the element is  $300 + (10 + 5 + 20) \times 2 = 370\text{px}$

# CSS font

- **font-family:** font-name|generic-family-name
  - A prioritized list of fonts to be used. The browser will try to use the first font if the it is available, and goes down the list
- The generic font family names include: serif (with small tails), sans-serif (without small tails), monospace, cursive, fantasy. Use monospace for program listing. Use sans-serif for computer display. serif are mainly used in prints (such as "Times" for newspapers and books)
- **font-size:** n|n%|xx-small|x-small|small|medium|large|x-large|xx-large|smaller|larger
- **font-weight:** normal|bold|bolder|lighter|100|200|...|800|900
  - You can use a number between 100 to 900, in multiple of 100. The value of 400 is the normal weight; while 700 is bold

# CSS font

- **font-style:** normal|italic|oblique
  - The italic uses italic font installed (some font families include the italic version); while the oblique is done by tilting the normal font
- **font-variant:** normal|small-caps
  - The small-caps is smaller than the uppercase
- **font:** style variant weight size/line-height family
  - Set all the font properties using a one-line shorthand notation. The properties must follow the order shown above. However, the leading and trailing items can be omitted



# CSS font

```
p {  
    font-size: 16px;  
    font-weight: bold;  
    line-height: 140%;  
    font-family: Arial, sans-serif;  
}
```

# Other CSS properties

- Text
  - Text-align, height, transform, etc.
- Background
  - Color, position, image, etc.
- Table
  - Border-spacing, table-layout, etc.

# CSS properties

- There are many properties and depending on which html tag you are applying to they may be interpreted differently
- Instead of memorizing all possible properties, use a small core set until it cannot do what I am looking for
- See <https://developer.mozilla.org/en-US/docs/Web/CSS/Reference> for a complete list

# Additional resources

- HTML: <https://www.w3schools.com/html/default.asp>
- CSS: <https://www.w3schools.com/css/default.asp>
- CSS+HTML: [https://www3.ntu.edu.sg/home/ehchua/programming/webprogramming/HTML\\_CSS\\_Basics.html](https://www3.ntu.edu.sg/home/ehchua/programming/webprogramming/HTML_CSS_Basics.html)