HTML and CSS

CSCI2720 2020–21 Term 2

Building Web Applications

香港中文大學
The Chinese University of Hong Kong

Dr. Chuck-jee Chau
chuckjee@cse.cuhk.edu.hk

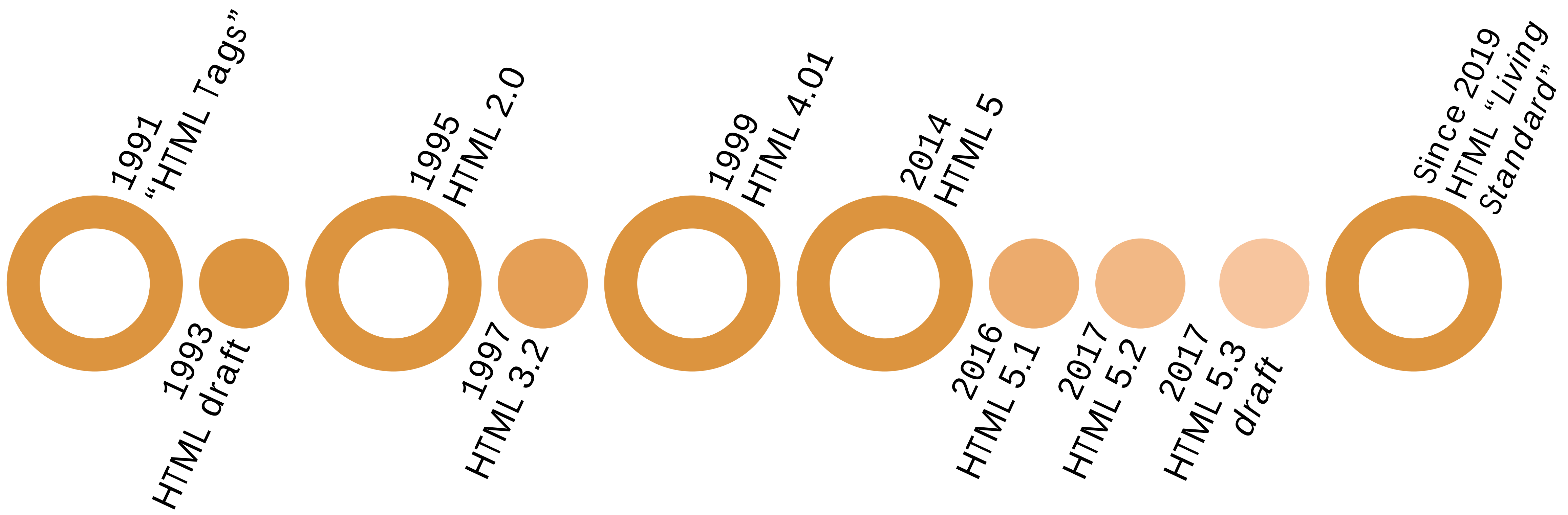
Outline

- HTML Basics
- Marking up elements
- Hyperlinks
- Encoding special characters
- CSS Basics
- Using CSS with HTML
- Selectors and properties
- Inheritance and cascading

HTML Basics

- HTML – Hypertext Markup Language
  - It is *not a programming language* but a rendering guideline for software
- The most *fundamental* code web browsers read to generate web contents

A brief history of HTML…



Some ideas on Web organizations...

- W3C *(since 1994)*
  - ***World Wide Web Consortium***
    - Founded by Tim-Berners Lee — the creator of WWW
  - Maintaining standards for WWW
    - Working draft, candidate/proposed recommendations
    - W3C recommendations

- WHATWG *(since 2004)*
  - ***Web Hypertext Application Technology Working Group***
  - Founded by people in leading web browser vendors

- Read: *https://www.w3.org/blog/2019/05/w3c-and-whatwg-to-work-together-to-advance-the-open-web-platform/*

Why HTML?

- HTML helps you to
  - *dedicate the roles* of text or media on the page
  - set up *hyperlinks* to allow navigation between pages
- HTML is well supported by web browsers on multiple device platforms, allowing a *unified experience*
- Although people rarely write HTML directly, you need to learn basic concepts to generate a page using *scripts*!

A simple HTML document

```
<!DOCTYPE html>
<html>
<head>
  <title>Welcome to CSCI2720</title>
</head>
<body>
  <p>This is a nice course, isn't it?</p>
  <!-- just kidding, nothing is easy -->
</body>
</html>
```

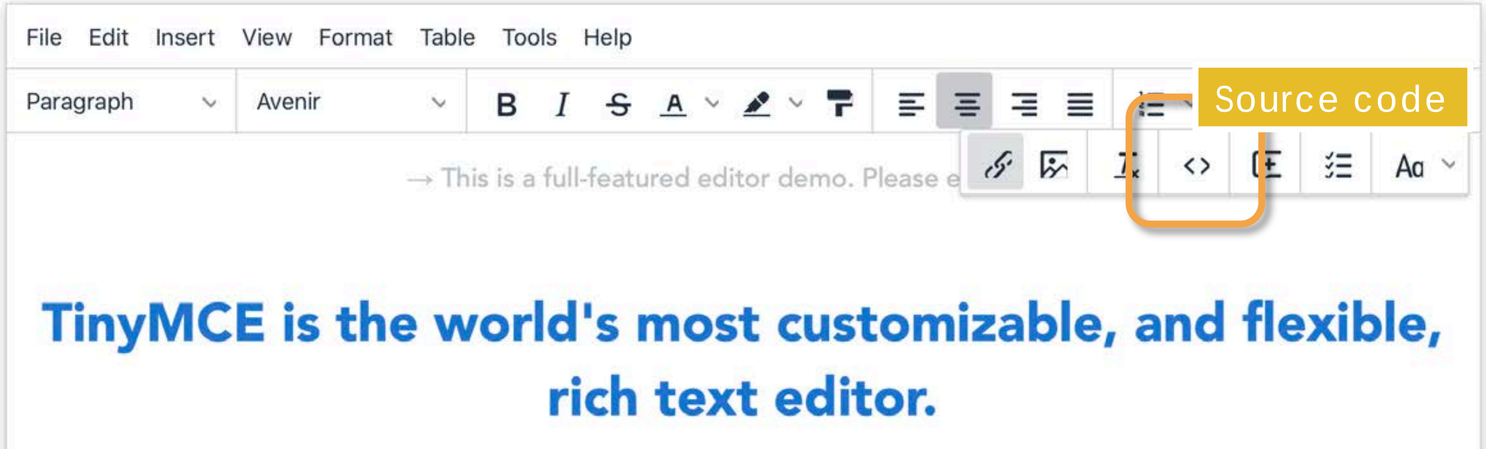
https://codepen.io/chuckjee/pen/abmjgVJ

- The `!DOCTYPE` declares the document type
  - "`html`" represents an HTML5 file
- The `<head>` section contains useful data but not for displaying, such as scripts and stylesheets
- The `<body>` section contains everything to be shown in the browser screen
  - `<!--` and `-->` denotes comments which will be ***ignored*** when rendering
- Usually this is saved as a .html file

Where do you see HTML code?

- Looking at the source code on any web page
  - *Chrome*: right click and choose *"**View Page Source**"*
  - *Safari*: right click and choose *"**Show Page Source**"*
- *"**Source code**"* in some WYSIWYG editors on web

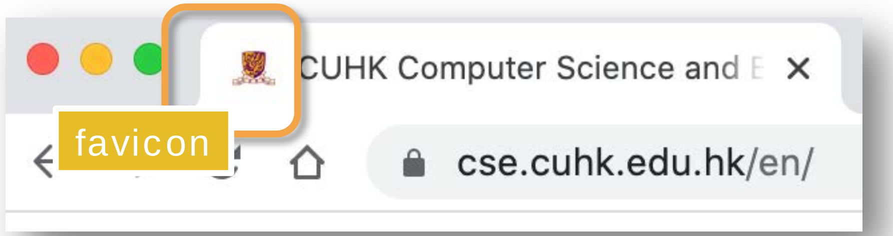
https://www.tiny.cloud



TinyMCE is the world's most customizable, and flexible, rich text editor.

The HTML head

- Some items are relevant to a web page, but are not contents to be shown in the page
  - Page title and "*favicon*" of a page
  - Stylesheets, scripts or other external files
  - Metadata like keywords for search engines to ***understand*** the page in their way

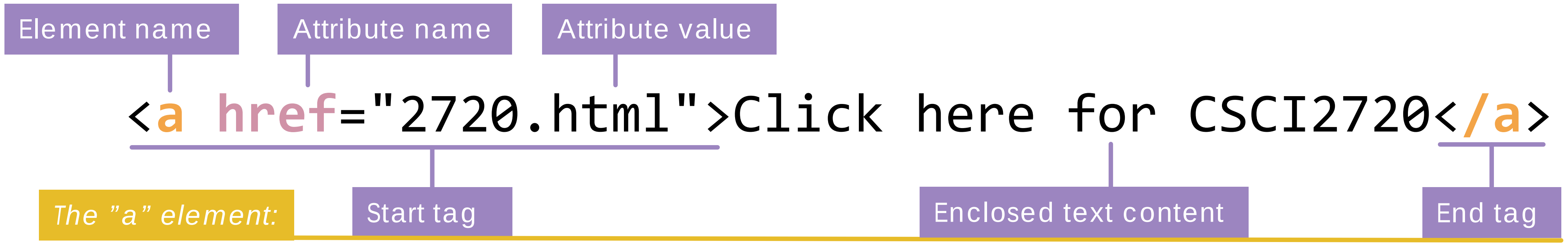


The HTML body

- All contents in the body will be shown in the page
  - *Paragraphs, headings, images, tables, …*
- You can also create a nice structure to present the contents in a semantic manner, using a *header*, *sections* and a *footer*
- Sometimes executable scripts are put at the end of the HTML body

The syntax of HTML elements

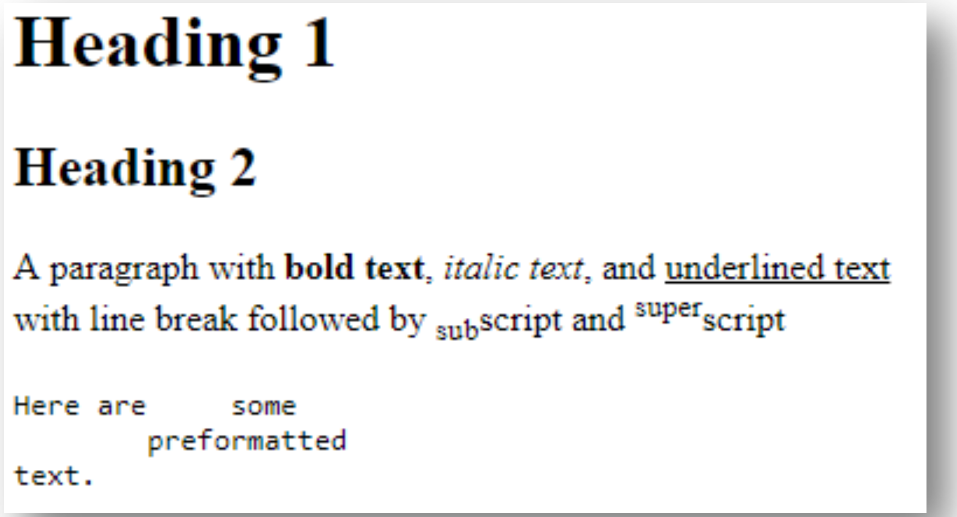
- All HTML elements building blocks of the web page
  - Whether they are shown or not
- Elements are created using tags in the code
  - Tags may or may not have an attribute
- Note: HTML is ***not case-sensitive***, yet recommended for small letter tags



```
<a href="2720.html">Click here for CSCI2720</a>
```

Marking up elements

- Headers `<h1>`, `<h2>`, …, `<h6>`
- Paragraph `<p>` and line break `<br>`
- Formatting
  - Bold `<b>`, italic `<i>`, underline `<u>`
  - Subscript `<sub>`, superscript `<sup>`
  - Pre-formatted `<pre>`



Heading 1

Heading 2

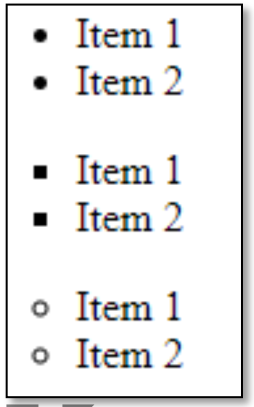
A paragraph with **bold text**, *italic text*, and underlined text
with line break followed by $_{\text{sub}}$script and $^{\text{super}}$script

```
Here are     some
        preformatted
text.
```

Marking up elements

- Lists
  - Ordered list `<ol>`
  - Unordered list `<ul>`
  - List items `<li>`

```
<ul>
<li>Item 1</li>
<li>Item 2</li>
</ul>

<ul type="square">
<li>Item 1</li>
<li>Item 2</li>
</ul>

<ul type="circle">
<li>Item 1</li>
<li>Item 2</li>
</ul>
```

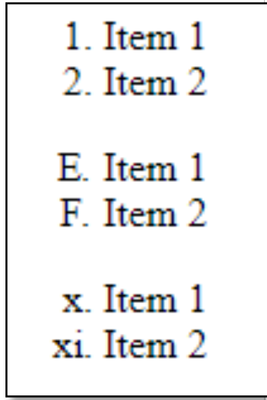


- Item 1
- Item 2

▪ Item 1
▪ Item 2

○ Item 1
○ Item 2

```
<ol>
<li>Item 1</li>
<li>Item 2</li>
</ol>

<ol type="A"
    start="5">
<li>Item 1</li>
<li>Item 2</li>
</ol>

<ol type="i"
    start="10">
<li>Item 1</li>
<li>Item 2</li>
</ol>
```

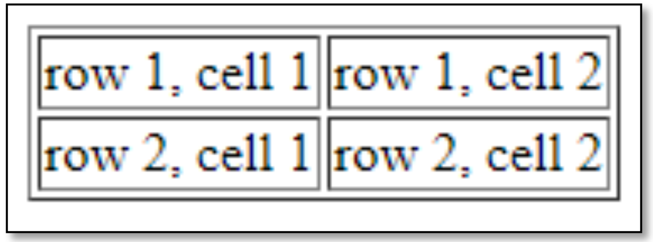


```
 1. Item 1
 2. Item 2

 E. Item 1
 F. Item 2

 x. Item 1
xi. Item 2
```

Marking up elements

- Tables `<table>`
  - A table is broken into rows `<tr>`
  - A row is broken into data cells `<td>`
  - Optional table header `<th>`

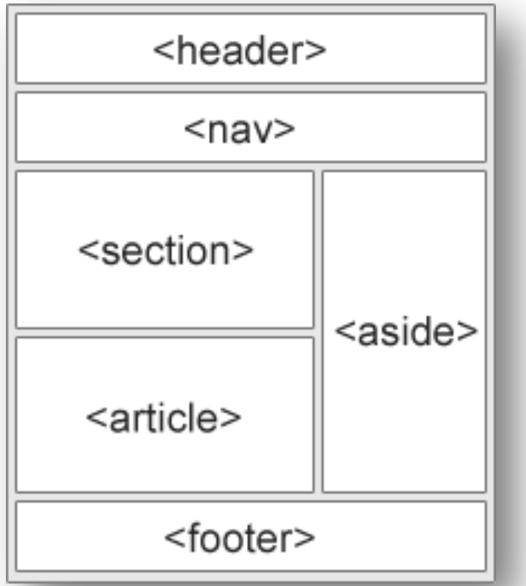
```
<table border="1">
  <tr>
    <td>row 1, cell 1</td>
    <td>row 1, cell 2</td>
  </tr>
  <tr>
    <td>row 2, cell 1</td>
    <td>row 2, cell 2</td>
  </tr>
</table>
```



row 1, cell 1	row 1, cell 2
row 2, cell 1	row 2, cell 2

Semantic elements

- In HTML5, it is recommended that the page contents are declared clearly into logical sections
  - E.g. sections `<section>`, navigation bar `<nav>`
- Browsers generally do not define how to render them
  - Easier for search engines and bots to know how data is organized on the page
  - Good anchor points for styling up the page with CSS
  - *Except bold* `<strong>`*, italic* `<em>`*, etc. which have predefined behaviours in a browser*
- *See: https://www.w3schools.com/html/html5_semantic_elements.asp*



Hyperlinks

- The hyperlink allows a "non-linear" manner of hypertext and hypermedia consumption
  - ***Inline links***: pointing to another file in the same server, or files on another web server
  - ***Anchors***: pointing to another part/section in the same file
- Usually displayed in different colours than normal text, depending on whether the link has been visited or not

Hyperlinks

- The `<a>` element
  - href attribute → what to point to
  - target attribute → where to open, e.g. **`"_blank"`** opens the link in a new tab/window
  - E.g. `<a href="csci2720.html" target="_blank">Webapp</a>` will open the files *csci2720.html* in a new tab/window
- Defining a fragment name using an id could be useful
  - E.g. We have `<h2 id="intro">Introduction</h2>` in *csci2720.html*
  - The link can point directly to it as:
    `<a href="csci2720.html#intro">Introduction</a>`

Absolute vs. relative paths

Absolute paths, e.g.

`http://www.cuhk.edu.hk/english/index.html`

- Using a complete URL (uniform resource locator)
  - Protocol (http)
  - Domain (www.cuhk.edu.hk)
  - Port (80, not typed by default)
  - Path (/english/)
  - Filename (index.html)

Relative paths

- Using the current document as reference
- E.g. We are at the address `http://www.cuhk.edu.hk/english/index.html`
- `<a href="../chinese/index.html">` brings us to the "chinese" directory under "../" parent directory
- What about `href="/index2.html"`?

Including images

- Modern browsers support generally lots of image types, usually using `<img>`
  - E.g. `<img src="cuhk.jpg" width="100" height="150">`
- *See: https://developer.mozilla.org/en-US/docs/Web/Media/Formats/Image_types*

- The special tag `<svg>` can be used for the Scalar Vector Graphics
  - Specifying contents of a graphic using elements, e.g. `<circle>`, `<line>`
- People also use `<picture>` for detailed control on responsiveness

Embedding audio and video

- The relatively newer elements of **`<audio>`** and **`<video>`** adds native multimedia support into browsers
- Since there are too many multimedia file formats out there, you can use multiple **`<source>`** tags to point to multiple files, e.g.

```
<video control width="500">
  <source src="2720ver1.mp4" type="mp4">
  <source src="2720ver2.webm" type="webm">
  <p>Your browser isn’t supported!</p>
</video> <!-- lines observed one by one -->
```

Encoding special characters

- Browsers doesn't like to see **<** or **>** in the text as they look too much like HTML tags, e.g.

```
<p>Hello, I believe x < y and y > z.</p>
```

  - Modern browsers may be able to guess correctly, but who want to risk losing some customers seeing your page?
- **<** should be typed as `&lt;` and **>** should be as `&gt;` in the HTML file
  - These are called "HTML entities" and there are a list of them
  - *See: https://dev.w3.org/html5/html-author/charref*

Handling space

- By default, more than one consecutive whitespace (space, new line, tab, etc.) in an HTML file will be regarded as one, e.g.
  - `<p>Hello      World</p>` will be rendered as:
    → Hello World
- `&nbsp;` is the "non-breaking space" for inserting multiple whitespace, or avoiding line breaks
- Whitespace is preserved in the <pre> environment

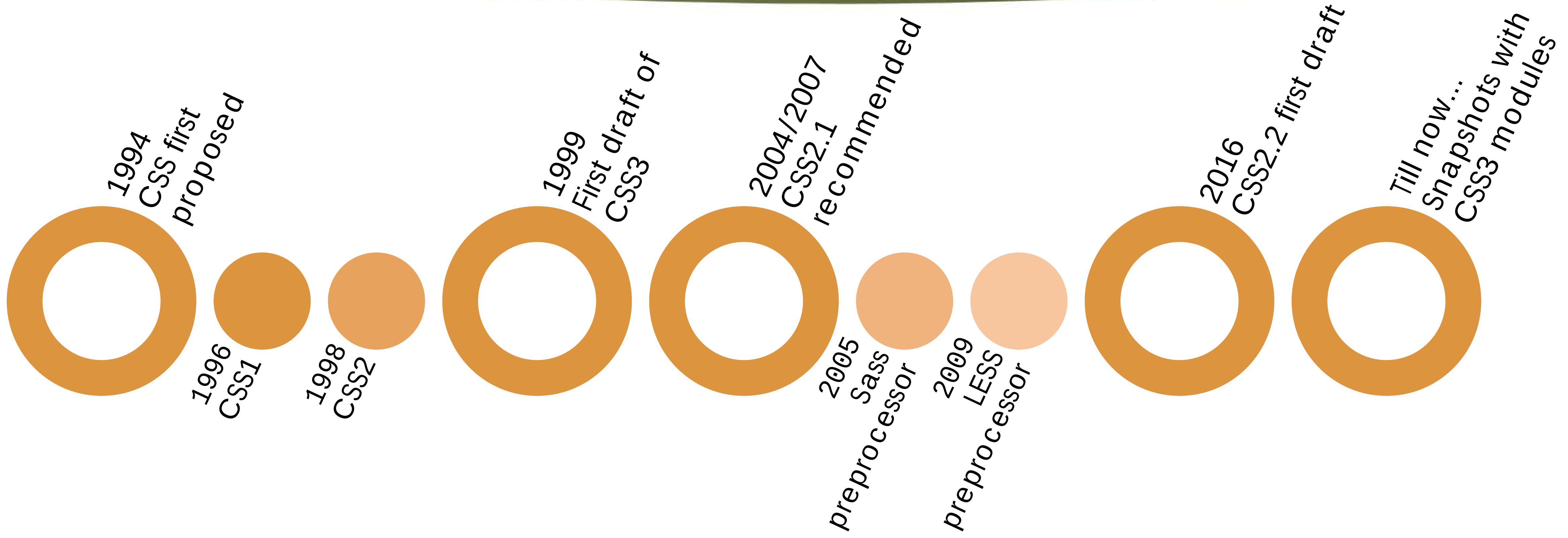
There are more to learn

- We have only gone through very superficial features in the HTML language and features
  - These are cornerstones which you will see again and again
- We will learn more throughout the course
- Check out HTML Cheatsheets, e.g.
  - https://www.wpkube.com/html5-cheat-sheet/

CSS Basics

- CSS – Cascading Style Sheets
- Again, it is ***not a programming language***, but is for styling contents in HTML

A brief history of CSS



Why CSS?

- Every element in HTML that are presentable has a set of style properties that can be modified via CSS
  - E.g. `font-family`, `color`, `line-height` of `<p>`
- Separating *design* from *contents*
  - Hopefully handled by different teams in development
  - Easily changing the skin of a web page
  - Sharing of the stylesheet among pages on the same site

CSS syntax

Selector

Declaration of property names with values

```
p {
   font-family: "Arial", "Helvetica", sans-serif;
   color: "Orange";
   line-height: 2em;
} /* a piece of comment, to be ignored by computer */
```

- Like HTML, CSS is generally not case-sensitive
  - Except HTML attribute values, e.g. the value of `id="SomeName"`

Using CSS in HTML

- If the task is to change the behaviour of `<p>` in an HTML file, there are multiple ways
  - *External stylesheet*: where a stylesheet file (.css) is linked
  - *Internal stylesheet*: the styles are included in the HTML head
  - *Inline styles*: specifying the behaviour for a particular tag directly using a style attribute
- More commonly, CSS could be created or changed using scripts to increase interactivity, changing link colours

External style sheet

- Include an external style sheet using **<link>** in **<head>**

```
<head>
...
<link rel="stylesheet" href="style1.css" >
...
</head>
```

```
h1 { text-align: center; font-family: Arial; }
h2 { color: #440000;
     text-align: center;
     font-family: Arial Black, Arial, Helvetica;
}
```

style1.css

Internal style sheet

- Putting a **<style>** tag inside **<head>**

```
<head>
...
<style>
  hr { color: sienna; }
  p { margin-left: 20px; }
  body { background-image: url("images/back40.gif"); }
</style>
...
</head>
```

Inline styles

- Set a style directly using a style attribute in the target tag

```
<p style="color: sienna; margin-left: 20px;">
This is a paragraph
</p>
```

Element and pseudo-element selectors

Element selectors	Description
`p`	Select all `<p>` elements
`h1, h2`	Select all `<h1>` and `<h2>` elements
`*`	Select all elements
`p a`	Select all `<a>` elements that is a child of a `<p>` element

Pseudo-element selectors	Description
`p:nth-child(3)`	Select all the `<p>` elements that are the 3rd child
`p::first-letter`	Select the first letters of all `<p>` elements

ID and class/pseudo-class element selectors

ID and class selectors	Description
`#example`	Select the only HTML element having attribute `id="example"` *Note: the id value should be unique in the document*
`.new`	Select all HTML elements having attribute `class="new"`
`p.new`	Select all `<p>` elements having attribute `class="new"`
`p a`	Select all `<a>` elements that is a child of a `<p>` element

Pseudo-class selectors	Description
`a:hover`	Select all `<a>` elements that has the mouse cursor over it
`a:link`	Select all unvisited `<a>` elements

Length units

- `em`
  - Relative to current font size
- `rem`
  - Relative to the root element font size
- `px`
  - One dot on screen (pixel)
- `%`
  - Size of the same property of the parent
- `vh`
  - 1% of the viewport (browser screen) height
- `vw`
  - 1% of the viewport width

- You can also use printed units like `cm` or `in`, yet results could be unexpected
- *See: https://engageinteractive.co.uk/blog/em-vs-rem-vs-px*

Some useful properties

- There are way too many properties you can set in CSS stylesheets
- Learn the useful properties and their possible values, and then look up new ones when needed!
  - Text: **`font-family`**, **`font-size`**, **`font-weight`**, **`color`**, …
  - Layout: **`text-spacing`**, **`line-height`**, **`text-align`**, …
- Want more?
  Read: https://css-tricks.com/lets-look-50-interesting-css-properties-values/
  - Some regarding page layout will be covered in the next lecture

Fonts

- Besides using installed fonts on the user's computer, you can also use web fonts with the `@font-face` selector
- There are popular online font repositories that you can use the fonts freely (*under certain licenses*)
  - E.g. *https://fonts.google.com*

Transforms, transitions and animations

- 2D and 3D transforms
  - `translate()`
  - `rotate()`
  - `skew()`
- Transition: specify a different `:hover` behaviour
- Animations: specify different behaviours for keyframes
- *See: https://learn.shayhowe.com/advanced-html-css/transitions-animations/*

Inheritance and cascading

- A child inherits (copies) the parent’s properties if unspecified
- The idea of “cascading” reflects priority of CSS rules:
  - More specific ones overrides generic ones
  - Inline styles overrides internal stylesheets, which overrides external stylesheets
  - Later ones overrides earlier ones
  - Properties marked !important overrides everything else

CSS Preprocessors

- For *easier and more efficient* web design
- More organized and cleaner code!
- Simplified work with variables, special selectors, etc.
- Source code to be compiled into regular CSS

```
$font-stack:    Helvetica, sans-serif;
$primary-color: #333;

body {
  font: 100% $font-stack;
  color: $primary-color;
}
```

Compile Sass into CSS

```
body {
  font: 100% Helvetica, sans-serif;
  color: #333;
}
```

CSS gurus

- CSS is very powerful to dramatically alter the appearance of a web page. There are simply too much that can be done!
  - Even rendering a “game”: CSS only Monument Valley *https://codepen.io/miocene/pen/NWRWQpX*
- You don’t need to learn everything. Know the syntax and learn reading the documentations!

Read further…

- HTML Living Standard
  - https://html.spec.whatwg.org
- w3schools.com HTML5 Tutorial
  - https://www.w3schools.com/html
- MDN HTML Guides and tutorials
  - https://developer.mozilla.org/en-US/docs/Learn/HTML
- w3schools.com CSS Tutorial
  - https://www.w3schools.com/css
- MDN Introduction to CSS
  - https://developer.mozilla.org/en-US/docs/Learn/CSS/Introduction_to_CSS

Checkpoint

How does an HTML document look?
How to understand some CSS properties?
How can I include CSS styles into an HTML file?