CM1040 Web Development Week 17 Lecture Note

Notebook:	Web Development		
Created:	2020-10-13 4:07 PM	**Updated:**	2020-12-19 12:31 PM
Author:	SUKHJIT MANN		

Cornell Notes	**Topic:** JavaScript template libraries	Course: BSc Computer Science Class: Web Development CM1040[Lecture] Date: December 17, 2020

Essential Question:

What are templating engines in JavaScript?

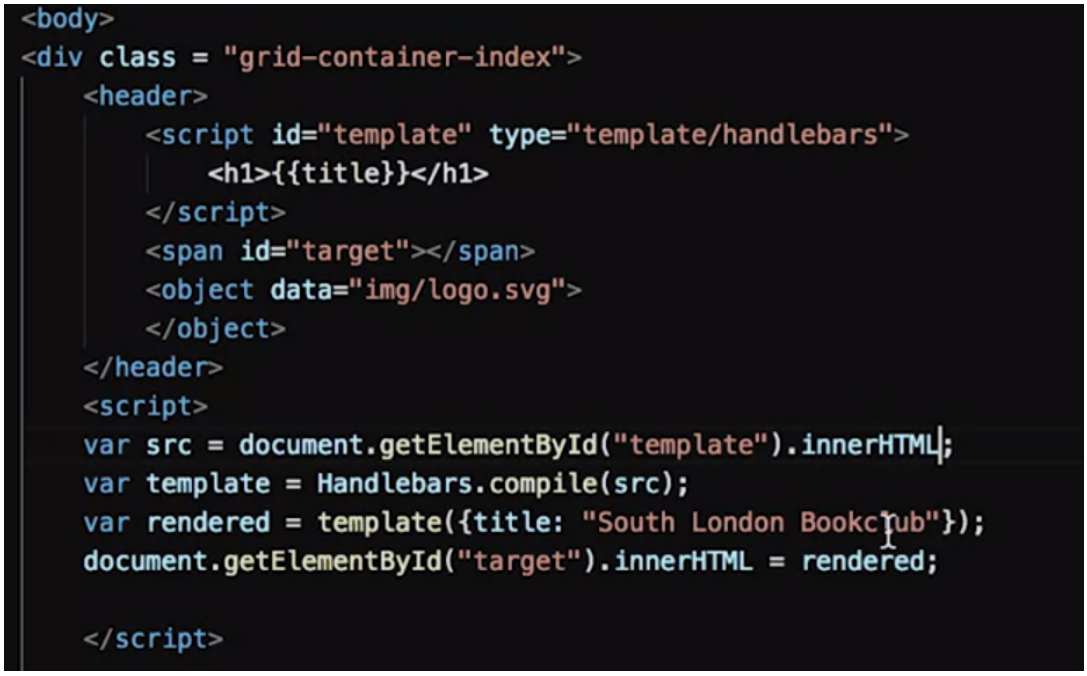
Questions/Cues:

- What is the use of web templates?
- How does sever-side rendering work?
- What is a template in terms of the Web?
- What is the reasoning behind why templating syntax appears the way that it does?
- What is Mustache?
- What is Handlebars.js?

Notes

- Web templates = are structured and well-laid out, its what the user sees. It separates the way the user sees the website from how the user uses the website, the functionality. HTML are structured and laid in exactly the same way as HTML pages.
    - Web developers work with templates to automatically generate web pages such as search result pages. They do this by creating web templates with special called placeholders. These placeholders are later replaced by dynamic data when the web templates are being executed or interpreted.
- Sever-side rendering = When visiting a site, the browser makes a request to a server which hosts the HTML pages for your website. The speed of the request depends on the speed of the internet, the place where the server is located, and how many other users are trying to access the server
    - Once the request is completed, then the browser is able to fully display the HTML, CSS and JS on the screen. The browser makes a request to the server for every you want rendered.
    - Sever-side rendering is good for one thing SEO or search engine optimization. SEO is the way we make a website appear on top of the search engine results, so more people can discover the website.
        - There are many ways for SEO on a website, one particular way is to create relevant content and allow the search engine crawlers, little bots that scour the internet for websites to find the content. When it comes to sever-side rendering, the site is on the server, so it's easy for the search engine crawlers to find it and index it

    - One way to avoid having to do both client-side and server-side templating is precompile client-side templates. Precompiling reduces the size of the templates, which allows for faster load times
- Template = in the case of a website, it's a model of an HTML that can derive or create less generic HTML pages
    - Has regular HTML tags but with placeholders, we later replace these placeholders with data
    - the reason placeholders look like {{name}} is dependent on the templating engine in use
        - The engine is what reads the template code or parses the template, and puts the data we provide in the position of these placeholders
        - Some templating engines are described as logic-less, this means they are limited in the way that they handle things like conditions or loops
        - There are some methods found across all templating engines such as, render() or compile()
            - Rendering is the process of combining the template code, the HTML, with the data
            - Compile translates the template into a JavaScript function, it takes the template apart, or parses it and then transforms it into JS. The JS function that is generated can then accept the data as a variable
- Mustache = purely logicless engine which can only replace quite specific placeholders with precomputed data. Mustache is clean, neat and tidy using a set of double curly on both ends of a value.
    - Another advantage of Mustache is that it is supported by many languages, this means that templates we create to processed in JS can at a later stage be processed in another language. This is useful if we want to create a site across many platforms that support different languages
- Handlebars.js = is a JS templating engine based on Mustache, it support most of the Mustache placeholders, but adds a lot of logic which makes it easier to use at times. Since it's still based on Mustache, this means that the handlebars.js templates we create work on Mustache processes \
    - The way we add logic in handlebars.js is by using helpers. Helpers easily add logic to the HTML templates while still keeping the code separate
    - handlebars template can either be precompiled for faster site load times or we can get the browser to compile them
        - <script src="handlebars.js"></script> or <script src=""></script>



```
<body>
<div class = "grid-container-index">
    <header>
        <script id="template" type="template/handlebars">
            <h1>{{title}}</h1>
        </script>
        <span id="target"></span>
        <object data="img/logo.svg">
        </object>
    </header>
    <script>
    var src = document.getElementById("template").innerHTML;
    var template = Handlebars.compile(src);
    var rendered = template({title: "South London Bookclub"});
    document.getElementById("target").innerHTML = rendered;

    </script>
```

Summary

In this week, we learned about client-side/sever-side rendering and web templates.