Embedding Styles and Scripts

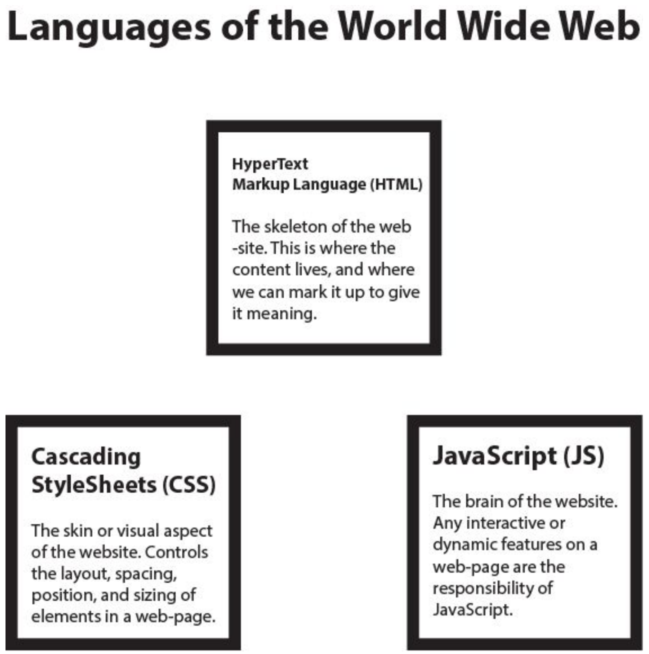
HTML5

Languages of the Web

As discussed previously in this course, there are three coding languages that make up websites on the internet:

- HTML(Hypertext Markup Language)
- CSS (Cascading Stylesheet)
- JS (JavaScript)

To help our projects remain organized and keep our code files separate on a language-by-language basis, we often keep CSS and JS files in subfolders so we—and the web browsers of our visitors—know where to look for them.


Languages of the World Wide Web
HyperText Markup Language (HTML)
The skeleton of the web-site. This is where the content lives, and where we can mark it up to give it meaning.
Cascading StyleSheets (CSS)
The skin or visual aspect of the website. Controls the layout, spacing, position, and sizing of elements in a web-page.
JavaScript (JS)
The brain of the website. Any interactive or dynamic features on a web-page are the responsibility of JavaScript.

Adding Style(s) to a Web Page

When it comes to dressing up the look and aesthetic of a web page, we must include CSS code. The best way to do this is to include a link element inside of your website's head.

The link element is self-terminating, so there is no need for a closing tag. It expects an "href" attribute with a value describing the path to the CSS file.

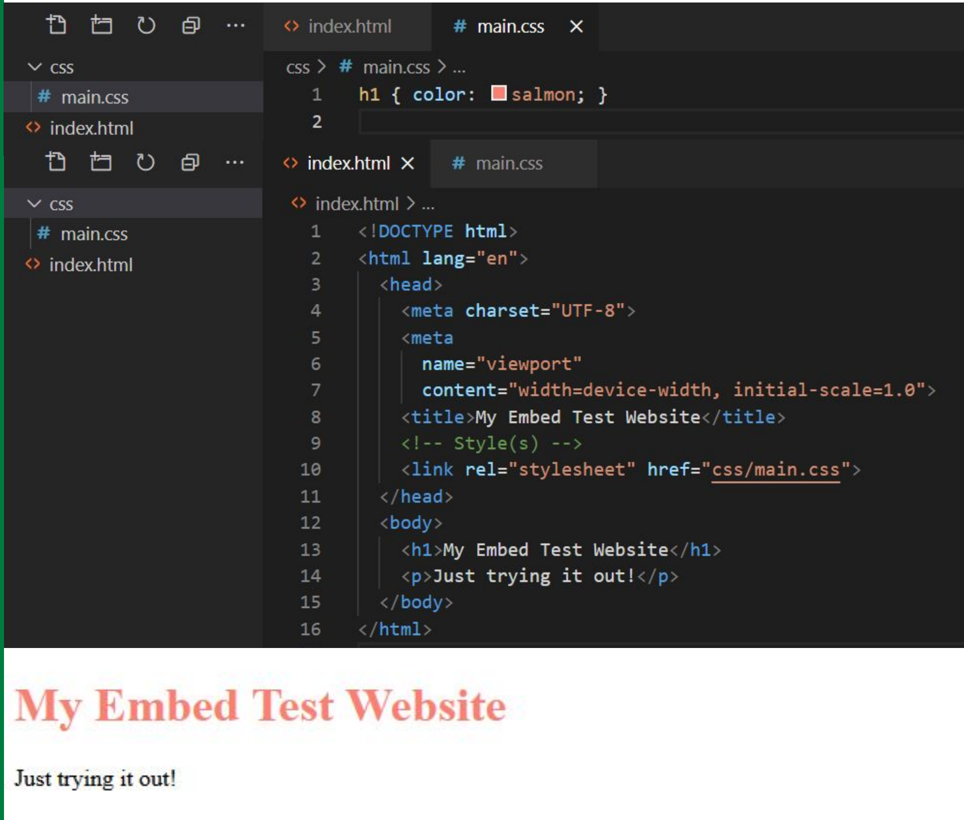
It is also important to include the "rel" attribute to let the browser know what type of file this is, as the link element can be used for relating various types of documents into your web page.

```
<link rel=”stylesheet” href=”folder/css-file.css”>
```

Try Linking a Stylesheet

Create a basic web page and stylesheet.

Once you have both a web page with a link element, and a basic stylesheet, try running it in the browser! Do you see the salmon text? If not, check the path!



```css
h1 { color: salmon; }
```

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta
      name="viewport"
      content="width=device-width, initial-scale=1.0">
    <title>My Embed Test Website</title>
    <!-- Style(s) -->
    <link rel="stylesheet" href="css/main.css">
  </head>
  <body>
    <h1>My Embed Test Website</h1>
    <p>Just trying it out!</p>
  </body>
</html>
```

My Embed Test Website

Just trying it out!

Other Ways to Inject Style(s)

It is recommended you utilize the link element for adding style to any given page, as it is far more organized than alternative approaches.

Note that other ways of adding and updating styling in a web page are cumbersome and typically very difficult to share across multiple pages. The link element also makes it easier for a web browser to load a page's content quickly regardless of included styles downloading, especially if a browser has CSS disabled.

The other two ways of including styles are:

1. **In-line Styles**
   You can add styles to any element in the body of your web page by adding a "style" attribute to the target element.

2. **Style Element**
   There is a style element capable of reading CSS code—simply add CSS code in between the opening and closing style tags.

Adding JavaScript(s) to a Web Page

As with styles, it is best to stay organized and separate your concerns into individual files.

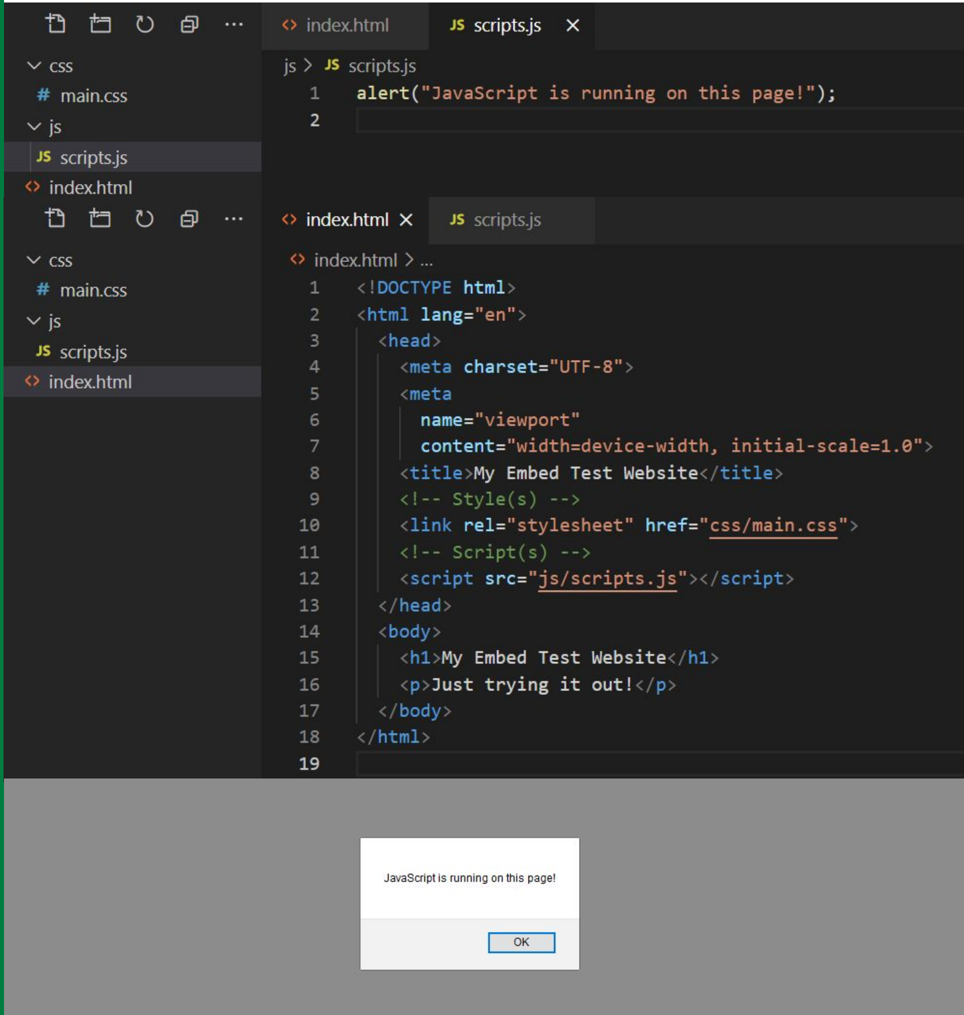
Instead of the link element, however, JS expects a script element with a “src” attribute value describing the path of the resource.

Note that the script element is not self-terminating, and must have both and opening and closing tag.

<script src=”folder/javascript-file.js”></script>

Try Adding a Script

Create a basic web page and JavaScript file. Check that the alert displays in your web browser!



```javascript
alert("JavaScript is running on this page!");
```

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta
      name="viewport"
      content="width=device-width, initial-scale=1.0">
    <title>My Embed Test Website</title>
    <!-- Style(s) -->
    <link rel="stylesheet" href="css/main.css">
    <!-- Script(s) -->
    <script src="js/scripts.js"></script>
  </head>
  <body>
    <h1>My Embed Test Website</h1>
    <p>Just trying it out!</p>
  </body>
</html>
```

JavaScript is running on this page!

Other Ways to Inject Scripts

Again, much like with CSS, there may be other ways to include scripting within your web page but this doesn't mean they are necessarily the right way. The recommended way is to use the script element and its src attribute to add scripts to your pages.

So you are aware there are a couple of other methods of adding JavaScript that you may come across…

1. **Inline Scripts**
   The reason that the script element opens and closes is that JS code can simply be included inside (if you omit the src attribute.)

```
<script>
  alert("There is JavaScript on this page!");
</script>
```

2. **JavaScript Attributes**
   There are a variety of JavaScript-specific attributes allowing for JavaScript event code (like a click) to be written as an attribute value.

```
<a href="#" onclick="alert('Link clicked!')">
  Click Here for Alert
</a>
```