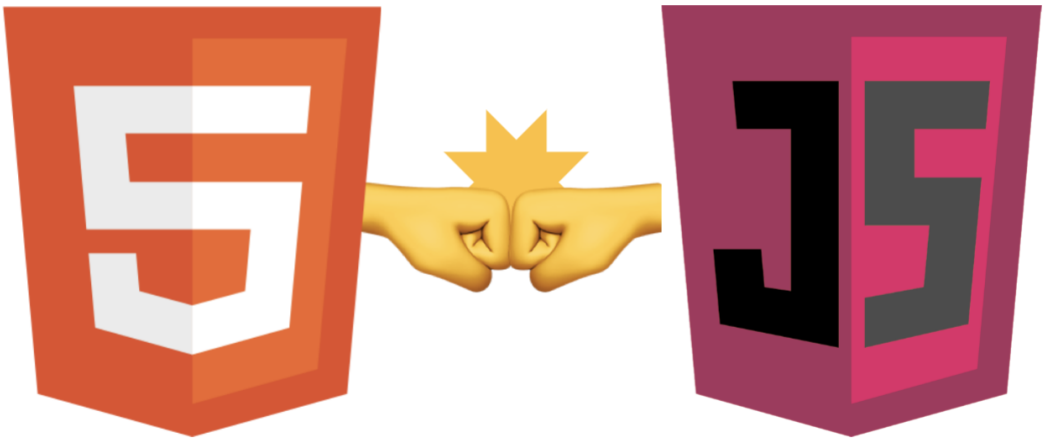
Placing JavaScript in HTML

In this lesson we'll see JS and HTML combine forces!
Let's begin!

WE'LL COVER THE FOLLOWING

- Listing 7-6.1: JS `alert`
- Listing 7-6.2: Deferred JS loading



In the previous chapter, you met the `<script>` tag that allows you to add inline or external JavaScript to the HTML page. Normally, when the browser reads the `<script>` tag, it immediately processes the JavaScript found there before reading the other parts of the HTML markup.

In Chapter 6, most scripts were either added just before the closing `</body>` tag or contained only function definitions that were invoked through the `onload` event attribute of the `<body>` tag to ensure that the content of the page was fully loaded before starting the execution of the script.

According to this behavior, the `alert()` method in this script is invoked

immediately when the browser processes the `<head>` section:

Listing 7-6.1: JS `alert`

```
<!DOCTYPE html>
<html>
<head>
  <title>Alert;</title>
</head>
<body>
  <h1>Display me!</h1>
  <script>
    alert("Hey, I'm running now...")
  </script>
</body>
</html>
```

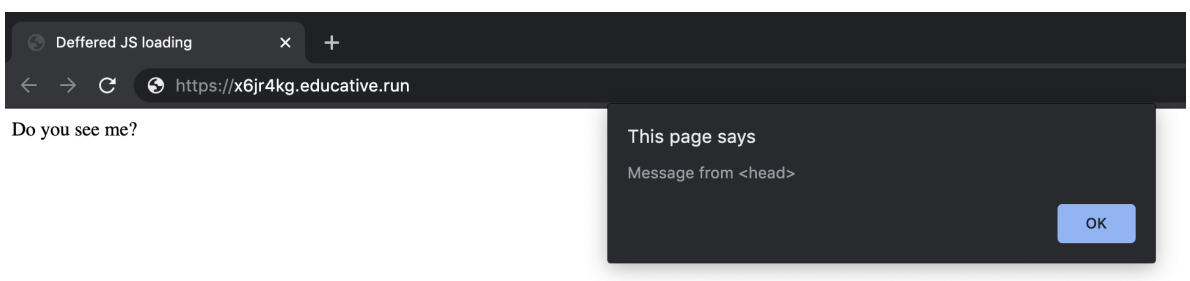
The `<script>` tag provides you two attributes, defer and async, which instruct the browser to change the normal way of processing the `<script>` content. Both attributes can be used only with external JavaScript files, otherwise, they do not have any effect.

Listing 7-6.2: Deferred JS loading

```
alert("Message from <head>");
```

According to the normal behavior, the index.html page should display the popup message before rendering any part of the page. However, the defer attribute in `<script>` tells the browser that the script will not be changing the structure of the page as it executes, and as such, the script can be run safely after the entire page has been parsed.

So, it will render the text in the document body before popping up the message, as shown in the image below:



The effect of the defer attribute

The `async` attribute instructs the browser that it should load the script **asynchronously** and execute the script at the first opportunity after download.

Scripts loaded with `async` do not guarantee that they wait for the loading of the full HTML document. If there are more `async` scripts, the order of their execution is not guaranteed, unlike defer, which guarantees that scripts will follow their definition order when executing them.

Show Useful Info

Let's get some insight into the basics of JS syntax in the next lesson.