Inline Frames

In this lesson, we will study inline frames in HTML.
Let's begin!

WE'LL COVER THE FOLLOWING

- Listing 5-10: Exercise-05-08/DisplayTitle.html
- Listing 5-11: Exercise-05-08/index.html

In Chapter 4 (Using Multimedia), you used the `<iframe>` tag to insert an inline video into a sample page. It's pretty easy to insert an inline frame into your web page, but before doing so, there are a few things you need to think over, and most of them are security considerations.

You know that you can add JavaScript code to your page, which runs when a web page is loaded, or as you interact with the page. This code can do many things in your browser, in fact it can do everything that the browser's current security configuration allows. If the inline page contains malicious code, that may run on your page.

You can use the sandbox attribute of `<iframe>` to specify extra security restrictions for the content of the inline frame. If you omit this attribute, your browser's default security restrictions are applied. When you set it to an empty string, all extra security restrictions are forced.

To constrain the inline frame in a more granular manner, you can add one or more restriction values to the sandbox, and separate these values by spaces. HTML5 defines the following constraints:

The `allow-scripts` enables the page to run scripts defined in the inline page. The inline content can submit forms only when allow-forms is set. With the allow-same-origin setting, the content is treated as being from the same origin as the containing document. By default, the frame's content can navigate only within itself, but applying the allow-top-navigation setting enables it to navigate content from the containing document.

In the source code download of this chapter, you will find a sample in the Exercise-05-08 folder. This project demonstrates the sandbox attribute. It contains a **DisplayTitle.html** file (Listing 5-10) that represents the content of an inline frame and the **index.html** file (Listing 5-11) which is a container that embeds frames.

Listing 5-10: Exercise-05-08/DisplayTitle.html

```
<!DOCTYPE html>
<html>
<head>
  <title>This is the title of this page!</title>
</head>
<body style="background-color: #dddddd">
  <button onclick="displayTitle()">
    Action!
  </button>
  <p id="output"></p>
  <script>
    function displayTitle() {
      var outputPar = document.getElementById("output");
      outputPar.innerHTML = document.title;
    }
  </script>
</body>
</html>
```

DisplayTitle.html has only a simple button. When you click it, the `displayTitle()` JavaScript function shows the title of this page.

Listing 5-11: Exercise-05-08/index.html

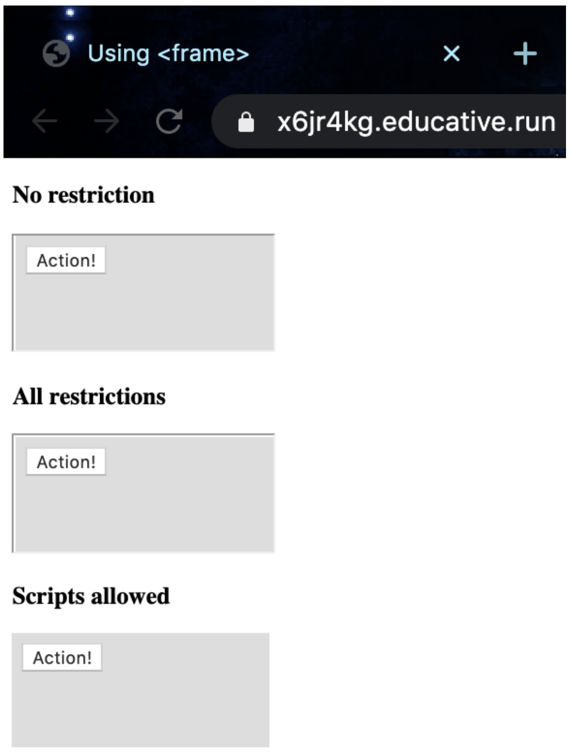
```
<!DOCTYPE html>
<html>
<head>
  <title>This is the title of this page!</title>
</head>
<body style="background-color: #dddddd">
  <button onclick="displayTitle()">
    Action!
  </button>
  <p id="output"></p>
  <script>
    function displayTitle() {
      var outputPar = document.getElementById("output");
      outputPar.innerHTML = document.title;
    }
  </script>
</body>
```

```
</html>
```

This page embeds the **DisplayTitle.html** file three times, with no restrictions, with all restrictions, and with allowing the scripts to run, respectively. When the browser renders this file, only the first and third frame outputs the title of the embedded page, because the second runs in a sandbox that does not allow running scripts.

Listing 5-11 demonstrates another feature of `<iframe>`. By default, it draws a border around the inline frame. This behavior can be turned off with the `seamless` attribute. However, `seamless` works only with Safari and Chrome.

With the little styling trick applied for the third frame, you can easily emulate `seamless`, as shown below:



Inline frame without the default border

In the *next lesson*, we will see how to define custom attributes.