Lesson Wednesday

Introduction to Programming (/introduction-to-programming) / JavaScript and Web Browsers (/introduction-to-programming/javascript-and-web-browsers) / Removing Event Listeners

Text Cheat sheet

Summary

- We must use a function declaration or a function expression that's stored into a variable when we create and remove an event listener.
- The two arguments we supply to the `removeEventListener()` method call MUST match the same arguments that we supplied to the `addEventListener()` method that originally created the event listener that we want to remove.
- The most common reason that you will remove an event listener is to reuse an HTML element in multiple scenarios/for different actions, or to react to an event only once.
- Issues related to scope that affect variables do not affect event listeners. We can remove an event listener in one function (or scope) that is originally defined in another function (or scope).

When and Why We Remove Event Listeners

We remove event listeners in three cases:

- When we only want an event handler to run once. We can remove an event listener after it has been run to make it so the reaction happens once.
- To reuse elements. Being able to add and remove event listeners allows us to use one element in multiple scenarios. For example, we could use one button in three different scenarios by adding and removing event listeners. The alternative would be to use three separate buttons with three separate event listeners.
- To improve performance or efficiency. If we have many, many event listeners in an application, removing the ones that are no longer needed can improve efficiency. This aspect of efficiency has to do with memory management. We won't learn about performance optimization at Epicodus, but this is an important topic to learn about down the road as you continue your growth as a developer.

Code Examples

```
function alertHeading() {
  window.alert("This is a heading element.");
}
let h1 = document.querySelector("h1");
h1.addEventListener("click", alertHeading);
h1.removeEventListener("click", alertHeading);  // this lin
e is new!
```

About the `removeEventListener()` method:

- This method takes two arguments:

    - The name of the event. In our example the event we're targeting is `"click"`.
    - The name of the function we are using to handle the event. This function is often called the "handler function". In our example, this handler function is `alertHeading`.
- The two arguments we supply to the `removeEventListener()` method call MUST match the same arguments that we supplied to the `addEventListener()` method that originally created the event listener that we want to remove.

(Some) Function Expressions Don't Work with `removeEventListener()`

We cannot use function expressions that are not stored into a variable as the second argument for the `removeEventListener()` method. In the following code snippet, we will fail to remove the event listener, even though the arguments in both `addEventListener()` and `removeEventListener()` are exactly the same. However, in JavaScript, anonymous function expressions that are not stored into a variable are not identical even if they are defined using the same unchanging source-code.

```
let h1 = document.querySelector("h1");
h1.addEventListener("click", function() {
  window.alert("This is a heading element.");
});
h1.removeEventListener("click", function() {
  window.alert("This is a heading element.");
});
```

Organizing Code into Multiple Functions

Anytime we organize our scripts into multiple functions, we'll follow the best practices listed below.

- All code that handles accessing or updating the DOM is considered user interface logic. Code that does not access or update the DOM is considered business logic (we'll see an example of this in an upcoming lesson).
- We don't need to define functions within the `window.onload` event handler. We only need to *call/use* the function inside of the `window.onload` event handler.
- We always place a function's definition above where we call on/use that function in our scripts.

Completed Mad Libs Example

With an advertisement that only runs once!

css/styles.css

```
.hidden {
  display: none;
}
```

mad-libs.html

```
<!DOCTYPE html>
<html lang="en-US">
<head>
  <script src="js/scripts.js"></script>
  <link rel="stylesheet" href="css/styles.css" type="text/c
ss">
  <title>A fantastical adventure</title>
</head>
<body>
  <h1>Fill in the blanks to write your story!</h1>
  <form>
    <label for="person1Input">A name</label>
    <input id="person1Input" type="text" name="person1Inpu
t">
    <label for="person2Input">Another name</label>
    <input id="person2Input" type="text" name="person2Inpu
t">
    <label for="animalInput">An animal</label>
    <input id="animalInput" type="text" name="animalInput">
    <label for="exclamationInput">An exclamation</label>
    <input id="exclamationInput" type="text" name="exclamat
ionInput">
    <label for="verbInput">A past tense verb</label>
    <input id="verbInput" type="text" name="verbInput">
    <label for="nounInput">A noun</label>
    <input id="nounInput" type="text" name="nounInput">
    <button type="submit">Show me the story!</button>
  </form>
  <br />
  <button type="button" class="hidden" id="reset">Reset</bu
tton>
  <div id="story" class="hidden">
    <h1>A fantastical adventure</h1>
    <p>
      One day, <span id="person1a">_________</span> and <sp
an id="person2a">_________</span> were walking through the
woods, when suddenly a giant <span id="animal">_________</s
pan> appeared. "<span id="exclamation">_________</span>", <
span id="person1b">_________</span> cried. The two of them
<span id="verb">_________</span> as quickly possible, and w
hen they were safe, <span id="person1c">_________</span> an
```

```
d <span id="person2b">_________</span> gave each other a gi
ant <span id="noun">_________</span>.
    </p>
  </div>
</body>
</html>
```

js/scripts.js

```
// User Interface Logic

function advertisement() {
  window.alert("Do you need a new computer? Visit www.super
extracomputersales.com to find the best deals!");
  document.querySelector("form").removeEventListener("submi
t", advertisement);
}

window.addEventListener("load", function() {
  let form = document.querySelector("form");
  let resetBtn = document.querySelector("button#reset");
  let story = document.querySelector("div#story");

  form.addEventListener("submit", function(event) {
    const person1Input = document.getElementById("person1In
put").value;
    const person2Input = document.getElementById("person2In
put").value;
    const animalInput= document.getElementById("animalInpu
t").value;
    const exclamationInput = document.getElementById("excla
mationInput").value;
    const verbInput = document.getElementById("verbInput").
value;
    const nounInput = document.getElementById("nounInput").
value;

    document.querySelector("span#person1a").innerText = per
son1Input;
    document.querySelector("span#person1b").innerText = per
son1Input;
    document.querySelector("span#person1c").innerText = per
son1Input;
    document.querySelector("span#person2a").innerText = per
son2Input;
    document.querySelector("span#person2b").innerText = per
son2Input;
    document.querySelector("span#animal").innerText = anima
lInput;
    document.querySelector("span#verb").innerText = verbInp
```

```
ut;
    document.querySelector("span#noun").innerText = nounInp
ut;
    document.querySelector("span#exclamation").innerText =
exclamationInput;

    story.removeAttribute("class");
    event.preventDefault();
  });

  form.addEventListener("submit", function() {
    resetBtn.removeAttribute("class");
  });

  form.addEventListener("submit", advertisement);

  resetBtn.addEventListener("click", function() {
    story.setAttribute("class", "hidden");
    document.getElementById("person1Input").value = null;
    document.getElementById("person2Input").value = null;
    document.getElementById("animalInput").value = null;
    document.getElementById("exclamationInput").value = nul
l;
    document.getElementById("verbInput").value = null;
    document.getElementById("nounInput").value = null;
  });
});
```

Previous (/introduction-to-programming/javascript-and-web-browsers/using-function-declarations-in-event-handling)
Next (/introduction-to-programming/javascript-and-web-browsers/practice-event-listeners)

disable dark mode

(http://www.epicodus.com)