Lesson Wednesday

Introduction to Programming (/introduction-to-programming) / JavaScript and Web Browsers (/introduction-to-programming/javascript-and-web-browsers) / Forms, Hiding and Showing Elements, and the Event Object

Text Cheat sheet

Summary

Here are the important takeaways:

- When creating forms, use a button element with a `type` attribute that's set to a value of `submit`. This button should always be inside of the `<form>` tags. With this HTML, we can use the `onsubmit` event handler property on the form to target the submission event.
- An `event` object is automatically created for every event handler when the event happens. We can only access the `event` object from within the event handler. The `event` object give us information and tools like:
  - What type of event it is (like 'click' or 'submission').
  - Where in the webpage the event originated from.

    - Stopping the default behavior for the event with the `preventDefault()` method.
- When we're working with a submission event on a form, we'll need to prevent the default behavior of refreshing the page. Use `event.preventDefault()` within the event handler and don't forget to add the parameter `event` to the function expression.
- `HTMLInputElement` s have a `value` property that we can access to get what the user has typed into the input.
- We can separate our UI logic into different functions to organize our code by its purpose.

About form `<input>` and `<label>` attributes

To create form fields with labels and inputs, we use `<label>` and `<input>` HTML elements. Each of these has a series of attributes that make the form function properly:

- The `id` attribute is the same `id` attribute that we already learned about. In this case, it helps us target and access that input's value.
- The `type` attribute in any `<input>` allows you to select the type of content that will go into that input. We've selected `text` because we're only inputting text. There are many, many values we can set for the `type` attribute, all of which change the look of the input element. We'll learn more about these in an upcoming lesson.
- The `for` attribute on the `<label>` element should have the same value as the `name` attribute on the corresponding `<input>`. This allows us to connect the label and input together in code. Functionally, this enables a user to click on the label to select the corresponding input (a text cursor will appear in the input). The `name` attribute on the `<input>` has other functions, too, which we won't get into now.

It's standard practice to include all of the above attributes on form `<label>` and `<input>` elements, in their respective locations.

Example: How to capture input when a form is submitted

HTML Form:

```
<form id="some-form">
  <label for="some-input">Your input:</label>
  <input id="some-input" type="text">

  <button type="submit">Submit!</button>
</form>
```

JavaScript to capture form information when form is submitted:

```
window.onload = function() {
  let form = document.querySelector("form");
  form.onsubmit = function(event) {
    event.preventDefault();
    const someInput = document.getElementById("some-inpu
t").value;
  };
};
```

Debugging Tips

- If you submit your form and then there's a `?` at the end of the URL in your address bar, you forgot to put `event.preventDefault();` , or you attached your event listener to the wrong form.

Completed Mad Libs Code

CSS

css/styles.css

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

HTML

mad-libs.html

```
<!DOCTYPE html>
<html lang="en-US">
<head>
  <link rel="stylesheet" href="css/styles.css" type="text/c
ss">
  <script src="js/scripts.js"></script>
  <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>

  <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 as possible, an
d when they were safe, <span id="person1c">_________</span>
and <span id="person2b">_________</span> gave each other a
giant <span id="noun">_________</span>.
```

```
    </p>
  </div>
</body>
</html>
```

Scripts

The following code does not include alternate forms of organization that the lesson covered!

js/scripts.js

```
window.onload = function() {
  let form = document.querySelector("form");
  form.onsubmit = function(event) {
    // in this section we get the value for each form input
    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;

    // then we set the story variables to the values we got
from the form
    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;

    document.querySelector("div#story").removeAttribute("cl
ass");
```

```
      event.preventDefault();
    };
  };
```

Previous (/introduction-to-programming/javascript-and-web-browsers/practice-event-handling-with-event-handler-properties)
Next (/introduction-to-programming/javascript-and-web-browsers/other-ways-to-organize-ui-logic)

Lesson 54 of 75
Last updated March 24, 2023

disable dark mode

(http://www.epicodus.com)