Lesson **Thursday**

Introduction to Programming (/introduction-to-programming) / JavaScript and Web Browsers (/introduction-to-programming/javascript-and-web-browsers) / Understanding Web APIs: Event Handling

Text

Let's continue to learn how to navigate Mozilla Developer Network (MDN) documentation and where to find more information about the Web APIs we learn about in this section. In this lesson, we'll learn about where to find more information about event handling and the Web API interfaces (object types) that contain the functionality for event handling.

We won't cover *all* the relevant object types for events — there's just too many out there! In this lesson we'll focus on just 3 topics:

- The `Event` object.
- Object types that inherit from the `Event` Object.
- The `EventTarget` object.

Finally, note that we'll review this information in future lessons. So if anything is not "clicking" now, there will be a chance to understand this information again.

The `Event` Object

Quite simply, the `Event` object represents an event that takes place in the DOM, like a form submission, scroll on a webpage, or click of a button. The `Event` object type is a generic object type, and there are many other object types that inherit functionality from `Event`. This means that the `Event` object contains functionality that's common to all events.

We're familiar with one method from the `Event` object:

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

Where `event` is an actual object — a variable that contains the data about the actual event that just happened. We only use `event.preventDefault();` to prevent the default action of a form submission event, which is to refresh the page. If you want a review about the `event` variable and `event.preventDefault();` visit the lesson Forms, Hiding and Showing Elements, and the `Event` Object (https://www.learnhowtoprogram.com/introduction-to-programming/javascript-and-web-browsers/forms-hiding-and-showing-elements-and-the-event-object).

The `Event` object type contains other helpful properties that can give us more information about the event that just occurred. Another commonly used property of the `Event` object is the `target` property (https://developer.mozilla.org/en-US/docs/Web/API/Event/target), which returns the element on which the event occurred. We use it like this:

```
event.target
```

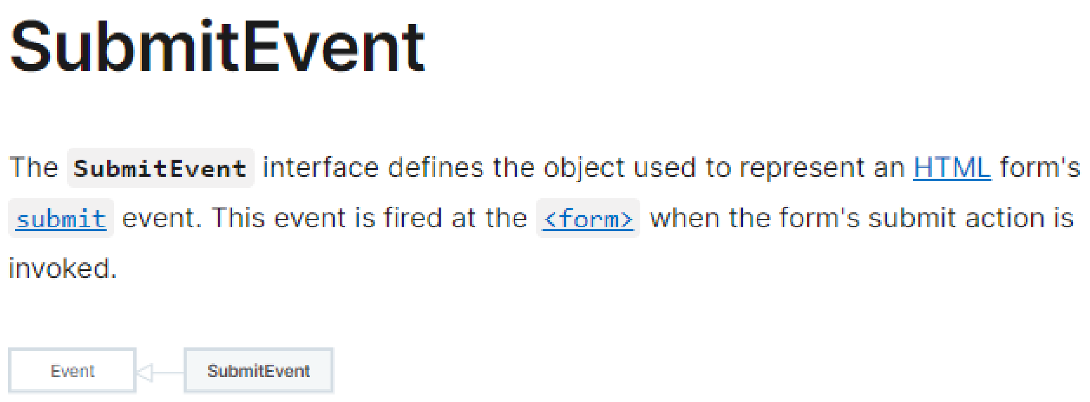
So if the event we're dealing with is a form submission, `event.target` would return the `<form>` HTML element.

To explore other properties and methods from the `Event` object, visit its documentation on MDN:

- The `Event` Interface (https://developer.mozilla.org/en-US/docs/Web/API/Event)

Object Types that Inherit Functionality from `Event`

As noted above, there are many object types that inherit from the `Event` object type. For example, when a form submission event happens, this event is represented by an object type called `SubmitEvent`. Take a look at the following image of the homepage of the `SubmitEvent` (https://developer.mozilla.org/en-US/docs/Web/API/SubmitEvent):



SubmitEvent

The **`SubmitEvent`** interface defines the object used to represent an HTML form's `submit` event. This event is fired at the `<form>` when the form's submit action is invoked.

The diagram in the above image communicates the chain of inheritance: `SubmitEvent` inherits from `Event`. This is just another example of two Web API interfaces that share functionality through inheritance. Many Web APIs do this and it's something to keep in mind as you use documentation online. If you want a more in depth review of inheritance, visit this lesson (https://www.learnhowtoprogram.com/introduction-to-programming/javascript-and-web-browsers/understanding-web-apis-interfaces-object-types-and-inheritance).

As far as exploring the various object types that represent specific events, like `SubmitEvent`, there's two good resources out there:

- MDN's documentation on events, where (most) events are listed by type and by its association with other interfaces. (https://developer.mozilla.org/en-US/docs/Web/Events)
- A section of the MDN docs on the `Event` objects that list object types that inherit from `Event`. (https://developer.mozilla.org/en-US/docs/Web/API/Event#interfaces_based_on_event)

The `EventTarget` Object

The last object type we'll learn about is the `EventTarget` object. `EventTarget` represents any object in our web browser that can be the target of an event. That includes `window`, `document`, any element within the DOM, and more! All of these objects have access to the functionality within the `EventTarget` object type because they *inherit* this functionality.

More specifically, `EventTarget` contains all of the methods that are used to set up **event listeners**. In total `EventTarget` has three methods, two of which we'll work with:

- `EventTarget.addEventListener()` (https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/addEventListener) which registers an event handler on the target of the event.
- `EventTarget.removeEventListener()` (https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/removeEventListener) which removes an event listener from the target of the event.

- `EventTarget.dispatchEvent()` (https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/dispatchEvent) which triggers an event to happen on the target of the event. We won't use this method in the program.

Each method has its own reference page that describes more information about use cases, syntax, and special considerations. To explore more about the `EventTarget` object, visit the MDN documentation:

- `EventTarget` on MDN (https://developer.mozilla.org/en-US/docs/Web/API/EventTarget)

More to Explore on MDN!

There are other benefits of using event listeners that are beyond the scope of what we're learning in this section. Optionally, visit the docs on the `addEventListener()` method (https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/addEventListener) if you want to take a deep dive to learn more.

The `addEventListener()` and `removeEventListener()` methods also have additional and optional parameters that you can explore in the docs. Some have to do with event bubbling and capturing, which we have not covered yet, so as always, don't worry about understanding everything in the reference.

Previous (/introduction-to-programming/javascript-and-web-browsers/practice-event-listeners)
Next (/introduction-to-programming/javascript-and-web-browsers/form-input-types)

Lesson 67 of 75
Last updated more than 3 months ago.

disable dark mode

Epicodus (http://www.epicodus.com)