1. Introduction

* What events are
* How to respond to events
* What data is stored in an event
* How to stop an event

When an event happens, the browser will make an announcement, and we can use JS to listen for the announcements and respond to them.

Fortunately, the Chrome browser has a special monitorEvents() function that will let us see different events as they are occurring.

// start displaying all events on the document object

monitorEvents(document);

// turn off the displaying of all events on the document object.

unmonitorEvents(document);

1. Respond to Events

<event-target>.addEventListener(<event-to-listen-for>, <function-to-run-when-an-event-happens>);

So an event listener needs three things:

1. an event target - this is called the target
2. the type of event to listen for - this is called the type
3. a function to run when the event occurs - this is called the listener

<https://developer.mozilla.org/en-US/docs/Web/Events>

1. Remove an Event Listener

Two objects that look the same are not exactly equal.

<event-target>.removeEventListener(<event-to-listen-for>, <function-to-remove>);

1. Quiz: Phases of an Event

Captures an event, at target, bubbles all the way back up.

Remember that the third argument of the .addEventListener() method is the boolean for the capturing phase. Since it's false, that means the event listener should not run during the capturing phase...therefore it will run in the (default) bubble phase.

1. Avoid Too Many Events

Started with 200 event listeners and unique functions, ended with one of each by using an event listener specific to the div.

1. Know When The DOM Is Ready

document.addEventListener('DOMContentLoaded', function () {

console.log('the DOM is ready to be interacted with!');

});

The target of the DOMContentLoaded event is the document object.

1. Lesson summary

Events are cool!

JS that modifies the DOM should be at the bottom of the HTML or circumvented with the DOMContentLoaded event listener.