FALL 2021 / ONLINE

INTERACTIVE DEVELOPMENT

Friday, October 15

WARM-UP QUESTION:

FAVORITE ANIMAL?

WARM-
FAVO L?

A05 RECAP

DEMOS + DISCUSSION

HARDEST PART OF JAVASCRIPT SO FAR?

LET'S
RECOVER
A FEW
THINGS.

THE EVENT LISTENER

The EventTarget method `addEventListener()` sets up a function that will be called whenever the specified event is delivered to the target.

`addEventListener()` works by adding a function or an object that implements EventListener to the list of event listeners for the specified event type on the EventTarget on which it's called.

```
// Add event listener to table
function  alertMe() {
        alert(“Hello”)
};
// Set variable for button
const button = document.getElementById("button");
// Add eventListener to button
button.addEventListener("click", malertMeodifyText, false);
```

Source: https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/addEventListener#add_a_simple_listener

THE EVENT LISTENER

The EventTarget method `addEventListener()` sets up a function that will be called whenever the specified event is delivered to the target.

`addEventListener()` works by adding a function or an object that implements EventListener to the list of event listeners for the specified event type on the EventTarget on which it's called.

```
// Add event listener to table
function  alertMe() {
        alert(“Hello”)
};
// Set variable for button
const button = document.getElementById("button");
// Add eventListener to button
button.addEventListener("click", alertMeodifyText, false);
```

Source: https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/addEventListener#add_a_simple_listener

TYPES OF EVENTS

Events are fired to notify code of "interesting changes" that may affect code execution. These can arise from user interactions such as using a mouse or resizing a window, changes in the state of the underlying environment (e.g. low battery or media events from the operating system), and other causes.

Each event is represented by an object that is based on the Event interface, and may have additional custom fields and/or functions to provide information about what happened.

```
// Set variable for form
const form = document.getElementById("form");

// Add eventListener to form
form.addEventListener("submit", ( )=> { alert(“Submitted”) }, false);
```

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

TYPES OF EVENTS

Different types of events:

- Events *(select)*
- Clipboard events *(copy)*
- Focus events *(blur)*
- Fullscreen events *(fullscreenchange)*
- Keyboard events *(keydown)*
- Mouse events *(click, mouseover)*
- Touch events *(touchend)*
- *…and many more!*

Source: https://developer.mozilla.org/en-US/docs/Web/API/Element#events

THE EVENT LISTENER

The EventTarget method `addEventListener()` sets up a function that will be called whenever the specified event is delivered to the target.

`addEventListener()` works by adding a function or an object that implements EventListener to the list of event listeners for the specified event type on the EventTarget on which it's called.

```
// Add event listener to table
function  alertMe() {
        alert(“Hello”)
};
// Set variable for button
const button = document.getElementById("button");
// Add eventListener to button
button.addEventListener("click", malertMeodi ext, false);
```

WHAT IS THIS?

https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/addEventListener#add_a_simple_listener

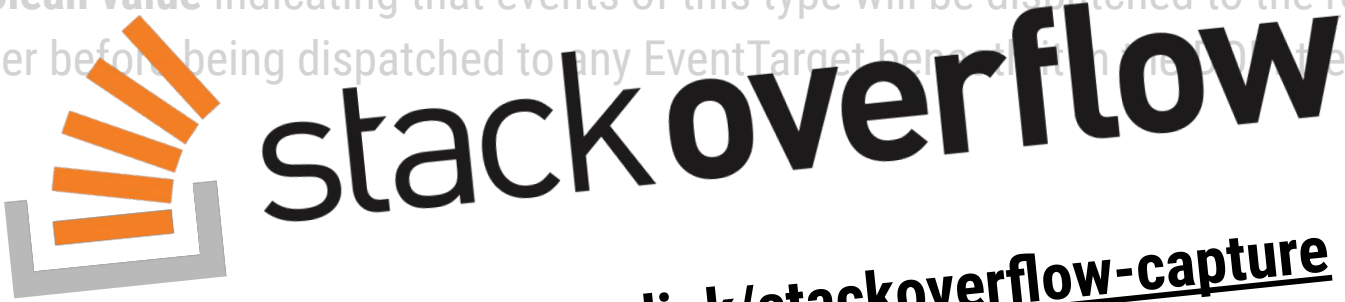
“CAPTURE”

A **boolean value** indicating that events of this type will be dispatched to the registered listener before being dispatched to any EventTarget beneath it in the DOM tree.

Source: https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/addEventListener#parameters

“CAPTURE”

A **boolean value** indicating that events of this type will be dispatched to the registered listener before being dispatched to any EventTarget beneath it in the DOM tree.t



https://maxx.link/stackoverflow-capture

“CAPTURE”

Events can be activated at two occasions: At the beginning ("**capture**"), and at the end ("**bubble**"). Events are executed in the order of how they're defined. Say, you define 4 event listeners:

```
window.addEventListener("click", function(){console.log(1)}, false);
window.addEventListener("click", function(){console.log(2)}, true);
window.addEventListener("click", function(){console.log(3)}, false);
window.addEventListener("click", function(){console.log(4)}, true);
```

The log messages will appear in this order:

- 2 (defined first, using capture=true)
- 4 (defined second using capture=true)
- 1 (first defined event with capture=false)
- 3 (second defined event with capture=false)

Source: https://stackoverflow.com/a/7398447

DEVTOOLS/WEB CONSOLE

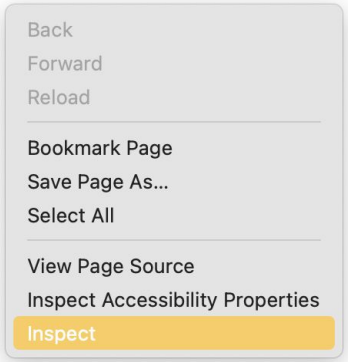
RECAP

All browser have some sort of developer toolkit to inspect the page.

- Chrome DevTools
- Firefox Web Console
- Safari Web Inspector
- Edge DevTools

You can open these from a menu, right-clicking on a webpage and clicking “**Inspect**”.

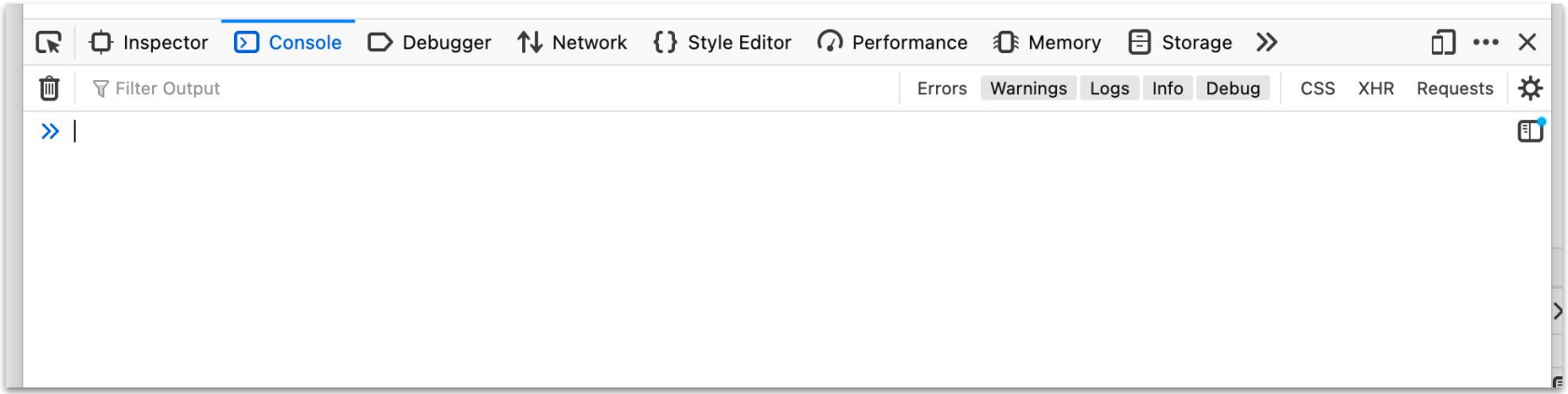
Open it up and click the “Console” tab.



DEVTOOLS/WEB CONSOLE

RECAP

Use the console in your browser (on a blank/new page) as a playground for the following walkthrough of code.



JAVASCRIPT ERRORS

Syntax errors

These are spelling errors in your code that actually cause the program not to run at all, or stop working part way through — you will usually be provided with some error messages too. These are usually okay to fix, as long as you are familiar with the right tools and know what the error messages mean!

Logic errors

These are errors where the syntax is actually correct but the code is not what you intended it to be, meaning that program runs successfully but gives incorrect results. These are often harder to fix than syntax errors, as there usually isn't an error message to direct you to the source of the error.

Source: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_went_wrong

CONSOLE.LOG

The `console.log()` method outputs a message to the web console. The message may be a single string (with optional substitution values), or it may be any one or more JavaScript objects.

```
console.log(“Hello World”);
// output:  Hello World

const button = document.getElementById(“button”);
console.log(button);
// output:  <button id="background" type="button">

console.log(typeof button);
// output:  object
```

Source: https://developer.mozilla.org/en-US/docs/Web/API/console/log

ERRORS DEMO

go to **https://maxx.link/javascript-errors**

ASSIGNMENT 06:
NUMBER GUESSING GAME

Follow the "A First Splash" MDN guide and create a number guessing game starting with this templated repo: https://github.com/maxxcrawford/number-guessing-game

- Besides creating the game, you will also:
  - Mobile response
  - Styled unique to you (Color, typeface, overall element designs)
- Check-in on Wednesday, Final due Friday, October 22, Beginning of class

Late submissions will be deducted -10% of the overall grade for each day the assignment is late unless arrangements are made with Maxx or Bryan.