Left off lesson 308

Recursion – Function calls itself (recursive case) until a base case condition is met.

\Variable Shadowing: when a variable declared within a certain scope has the same name as a variable declared in an outer scope.

Closure: Closes over surrounding variable environment.

Every function is a closure. Each function has its own lexical environment, including global and any other outer lexical environments. Latest or reassigned values of lexical variables are used.

Lexical environments or scopes are created by functions, for loops, if statements and stand alone curly braces.

Factory function creates another function.

Capturing / Bubbling - Order of Event listener triggering- Bubbling is the default

event.target- Access DOM element

i.e. - event.target.disabled = true – To disable a button that was clicked

Event object is created by the browser and then is passed into the handler function

window.location, or just location

location.href=’webPage URL’ – navigate to this page

navigator.clipboard

nlocation.geolocation.getCurrentPosition(() => {})

location.assign(‘webPage URL’) – navigate to this page (same as href above)

location.replace(’webPage URL’) - can't go back

location.host, location.origin, location.pathname – Current location varieties

window.history, or just history

history.back () – goes back one page

history.forward() –

history.go(#pages back) –

history.length – Number of pages browsed in current tab.

Execution context: 1) Variable environment, 2) Scope chain, 3) this keyword

This will keep old prototype data:

Object.setPrototypeOf(object, { …Object.getPrototypeOf(object), new prototype properties})

in practice all browsers use [\_\_proto\_\_](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/proto). The standard way to access an object's prototype is the [Object.getPrototypeOf()](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/getPrototypeOf) method. in practice all browsers use \_\_proto\_\_. The standard way to access an object's prototype is the Object.getPrototypeOf() method.

The prototype is itself an object, so the prototype will have its own prototype, making what's called a prototype chain. The chain ends when we reach a prototype that has null for its own prototype.

WEB APIs

Asynchronous tasks run in WEB API

Promises, Timers, AJAX Calls, Loading Images, ie. el.src = ‘image.jpg’;

Callback functions for asynchronous tasks are registered with the WEB API - to be later loaded in the callback queue or microtask queue.

Callback queue also contains callbacks for DOM events listeners. With the exception of load events, most of these listeners are not asynchronous tasks though.

Event loop tick: Event is loaded from callback queue to call stack.

All promises (built or from a fetch, etc.) use the microtask queue. An example of such a promise is fetch(‘<https://someurl.com/api>’).then(res => console.log(res))

Microtask queue can starve the callback queue.

Left off 299 28:28

Aria – opacity: 0 – screen readerwill still read element, need display: none.

=== operator does not perform type coercion as with ==

Default exports usually used when only one thing is to be exported from a module. Exports a value (such as an anonymous function) instead of a named variable.

getComputedStyle().stylepropertyname – Gets runtime property values of element styles.

Can access relative and absolute urls.

Document.documentElement – selects entire document. If I want to apply styles, etc. on a global basis.

Document.head – selects head.

Document.body – selects body.