W03 - Events

- Introduction
- Adding Event Listeners
- The Event Object
- Mouse, keyboard, and touch events
- Removing Event Listeners
- Stopping default behavior
- Event Propagation

```
document.body.addEventListener("click", doSomething);
```

The page will run as normal until the click on the page happens
→ click events include mouse, Enter key, screen tap

Inline Event Handler - `onClick="Something"` <- example | Not Recommended?

Older Event Handlers - `document.onClick = function() ...`
a single function can be used for each event

Using Event Listeners - Recommended, can add multiple functions independently to different events.

addEventListener() method is called on node objects.

Parameters - 1. type of event
2. callback function
3. Boolean to signify whether to capture or not (Default is false)

add a click event listener to whole page:

```
addEventListener('click', () => alert('You Clicked!'));
```

↑ a named function can be declared and referenced
- Parentheses aren't used to call the named function.

The Event Object - when triggered the event Listener passed the function as an event object.

Types of events - type property returns the type of event that occurred.

Event target - target property - reference node that fired event

Coordinates of Event - screenX, screenY
clientX, clientY
pageX, pageY
} return number of pixels from top right of specified box

W03 - Events

Types of Events → full list: https://developer.mozilla.org/en-US/docs/Web/Events

Mouse Events

mousedown mouseup

const dblclickParagraph = 'dblclick' } avoid using both on same element, since it is difficult to tell if a click is a single click or first of a double click
'click' }

'mouseover' }
'mouseout' } fired when mouse over specified element on page
'mousemove' }

Keyboard Events - keydown, keypress, keyup - When user presses a key, the event happen in that order

keydown occurs when a key is pressed and continues as long as the key is held down

keypress occurs after keydown but before keyup. Only occurs for keys that produce character input and delete.
- the most reliable way to find out the specific character that was pressed.

keyup occurs when key is released

Modifier Keys

event.ctrlkey
.shiftkey
.altkey
.metakey
} checks if modifier key was held down during the event.

Touch Events - smartphones, tablets, touch screen monitors...

touchstart - initial touch
Touchend - when user stops touching surface
touch move - after user touches screen then moves
touch enter - when user touches screen then moves into an element
touch leave - when user is still touching device and leaves element
touchcancel - touch event is interrupted

Touch event properties

touches - list of touch objects representing all touches taking place
↳ length - how many touch points, access using index notation
·touch.screenX, touch.screenY ·touch.force - pressure between 0-1
·touch.radiusX, touch.radiusY

→ Touch Properties are still experimental and not widely implemented

W03 - Events

Removing Event Listeners - removeEventListener()

after function is called, the listener can be removed.

Using an anonymous function as an argument to addEventListener() will prevent removing it later. There needs to be a reference to the same function name in the arguments of removeEventListener().

Stopping Default Behavior - preventDefault()

i.e. Prevent redirects to url links

```
<a id='broken' href='https://sitepoint.com'> Broken Link </a>


const brokenLink = document.getElementById('broken');

brokenLink.addEventListener('click', (event) => {
    event.preventDefault();
    console.log('Broken Link!');
});
```

preventDefault() should rarely be used. Some events cannot be overridden. cancellable property will return false if it cannot be overridden. defaultPrevented property will return true if default behavior has been overridden.

Event Propagation -

```
<ul id='list'>
    <li> one </li>
    <li> two </li>
    <li> three </li>
</ul>
```

when clicking on one of the <li> elements, also clicking on <ul> and <body> and <html> elements. An event propagates as it moves from one element to another

Propagation is the order that the events fire on each element.

- bubbling
- capturing

developers can decide which they want to use

Bubbling - element clicked first, then bubbles up the document tree

Capturing - root element first, then propagates downward.

W03 - Events

Bubbling - Default behavior

click on <li> with an event listener will first show, then bubble up to the <ul> with an event listener.

click on <li> without event listener will bubble up to the <ul> with event listener

Capturing - may want events on outer elements to fire before any events on the element that was actually clicked.

Stopping the Bubbling Phase - event.stopPropagation()

→ using stopPropagation() method may stop event listeners further up the chain from firing

Event Delegation - event listener attached to a parent element to capture events triggered by its children.

→ a single listener can be used for all children, using the target property to identify the property that was clicked.

→ All children inherit the event listener, so adding a child does not adding an additional event listener.