Module 4 Day 4

Event Handling

Event-Driven Programming

- The browser recognizes when *anything* happens on a page
  - Mouse click, mouse over, input field change, form submit, and on and on…
  - These are all *events*
- Every event occurs (is *triggered*) on a *target* (DOM) element
  - The event is "published"
- Our JS code *asks* to be notified for specific events
  - This is called "subscribing to an event", or
  - Adding an event listener
- An *event listener* is JS code that we write (a function)
- This is called an *event-driven interface*
  - The user determines the flow

Subscribing to an Event

- element.addEventListener(eventName, eventHandlerFunction)

```
// We got an element to subscribe to. Hook up the events.
element.addEventListener('mousemove', (ev) => {
  LogEvent(ev);
});
```

Events

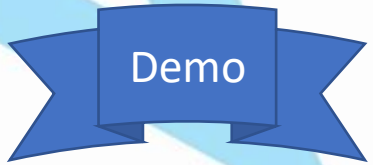
- MouseEvent
  - click, dblclick, mouseover, mouseout, mousemove
- KeyboardEvent
  - keypress, keyup, keydown
- Event
  - change (input, select or textarea), submit (form), reset (form)
- FocusEvent
  - focus, blur

Event Object

Event type	Properties
Event	target (an element), type (e.g., 'click', 'blur') preventDefault() method
UIEvent : Event	Parent event for mouse, keyboard, focus and other event types. No properties of interest to us at this time
MouseEvent : UIEvent	clientX, clientY, shiftKey , altKey, ctrlKey, button
KeyboardEvent : UIEvent	key, shiftKey , altKey, ctrlKey, repeat https://developer.mozilla.org/en-US/docs/Web/API/KeyboardEvent/key/Key_Values
FocusEvent : UIEvent	No fun properties

Page Loading Sequence

- Browser reads the page and starts processing elements top-down
  - As it is read, browser builds the DOM
  - Page also may start to render
- When a <script> tag is encountered, the browser stops other processing and runs the script
- When entire page is read and the DOM is built, document.DOMContentLoaded is triggered
- Browser continues to get external files (CSS, IMG) to complete the page
- When all external content has been loaded, window.load is triggered
- https://javascript.info/onload-ondomcontentloaded
- https://www.innoq.com/en/blog/loading-javascript/



Adding Event Handlers (Listeners)

- Add a handler to document.DOMContentLoaded event
  - This event fires when the HTML has been downloaded and parsed
  - Meaning all DOM elements exist in the tree
  - Page will not have rendered yet
  - External resources (css, jpg) may not have been downloaded yet
  - This code should be in global scope (not within another function)
- In that handler, add other handlers

```
document.addEventListener("DOMContentLoaded", () => {
    // Register all of your event listeners here
});
```

Event Bubbling (Propagation)

- An event is triggered on some source element
- Browser looks for event handler on the element, invokes if found
- Then it keeps looking for event handler on the element's parent, invokes if found
- And so on, up to the window object
- If you want to change this and stop the "bubbling", call event.stopPropagation()

Preventing Default

- Anchors <a> and Submit buttons <input type="submit"> have default behavior
  - Anchor navigates to a URL when clicked
  - Form posts to server when submitted
  - Clicking a checkbox toggles the checked state of the control
- You may want to override their behavior
  - E.g., use an anchor to hide a section
- To prevent the default behavior from happening, call event.preventDefault()
- NOTE: preventDefault does not stop propagation